Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

Жаріков Едуард В'ячеславович

УДК 004.75

ДИСЕРТАЦІЯ

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ УПРАВЛІННЯ IT-ІНФРАСТРУКТУРОЮ ХМАРНОГО ЦЕНТРУ ОБРОБЛЕННЯ ДАНИХ

Спеціальність: 05.13.06 – інформаційні технології

Подається на здобуття наукового ступеня доктора технічних наук
Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____ Е. В. Жаріков

Науковий консультант
Теленик Сергій Федорович
доктор технічних наук, професор

Київ – 2020

АНОТАЦІЯ

Жаріков Е. В. Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора технічних наук за спеціальністю 05.13.06 – інформаційні технології. – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, 2020.

У дисертаційній роботі вирішено науково-практичну проблему забезпечення ефективного функціонування ІТ-інфраструктури хмарного ЦОД в умовах невизначеності і змінних навантажень через створення методології управління та на її основі розроблення і застосування інформаційної технології з метою надання хмарних послуг із заданими показниками якості кінцевому користувачеві. Запропонована інформаційна технологія забезпечує ефективне функціонування ІТ-інфраструктури ЦОД провайдера хмарних послуг та на відміну від відомих базується на методології управління ІТ-інфраструктурою, враховує суттєві характеристики хмарних обчислень, гетерогенність та ієрархічність ІТ-інфраструктури, нові архітектури організації обчислень, а також використовує адаптацію до непередбачуваних навантажень за рахунок прогнозування, що дозволило забезпечити виконання заданих вимог угоди про рівень обслуговування та зниження операційних і капітальних витрат.

Розроблена інформаційна технологія базується на методології управління ІТ-інфраструктурою хмарного ЦОД, яка об’єднує такі розроблені положення, підходи, моделі і методи: операторну форму постановки, аналізу і розв’язання задач управління ІТ-інфраструктурою хмарного ЦОД; визначення і застосування виділених стратегій управління ресурсами і навантаження хмарного ЦОД, а також схем їх реалізації; декомпозицію хмарної ІТ-інфраструктури на три рівні (інфраструктури, платформи та застосунків); врахування традиційної, конвергентної і гіперконвергентної архітектур сучасних хмарних ЦОД; застосування адаптивного методу комбінованого прогнозування навантаження для визначення керуючих впливів на ІТ-інфраструктуру хмарного ЦОД;

застосування методу інтегрованого управління гетерогенними ресурсами ЦОД з урахуванням порушень SLA, споживання електроенергії і необхідної потужності на наступному кроці управління; застосування методів стохастичного пошуку (променевий пошук, відпал, навчання з підкріпленням) для реалізації визначених стратегій управління ІТ-інфраструктурою хмарного ЦОД; застосування методу управління розподіленим дворівневим сховищем з реплікацією для гіперконвергентних систем; врахування нових метрик стану ІТ-інфраструктури (миттєвий і середній коефіцієнти життєздатності ВМ, індикатор дисбалансу ФС, коефіцієнт відношення необхідних ресурсів до середнього об'єму наявних ресурсів, поріг вільних ресурсів) для визначення поточної стратегії управління; урахування програмно-визначених контролерів управління трьома основними ресурсами ЦОД (обчислення, сховище, мережа); використання комбінації централізованого управління за допомогою глобального менеджера ЦОД і децентралізованого управління на рівні ФС в залежності від обраної стратегії управління на поточний момент часу.

Для вирішення проблеми ефективного управління ІТ-інфраструктурою хмарного ЦОД і розв'язання поставлених задач використані: теорія систем, методи теорії ієрархічних систем, методи математичного програмування, методи дослідження операцій і теорії прийняття рішень, методи математичного та імітаційного моделювання, методи теорії штучного інтелекту, стохастичні і евристичні методи пошуку, методи прогнозування, методи математичної статистики сервісні моделі хмарних обчислень. Достовірність та обґрунтованість отриманих результатів обумовлені коректним використанням математичного апарату, а також підтверджуються результатами обчислювальних експериментів.

Наукова новизна одержаних результатів визначається такими теоретичними і практичними результатами, отриманими автором:

1) вперше розроблено методологію управління ІТ-інфраструктурою хмарного ЦОД на основі операторної форми постановки, аналізу і розв'язання задач управління в умовах невизначеності і змінних навантажень, яка відрізняється вибором стратегій управління в різних режимах роботи хмарного

ЦОД за рахунок визначення відповідних моделей через комбінування критеріїв і обмежень, а також за рахунок визначення відповідних схем реалізації, що дозволяє при управлінні хмарним ЦОД перейти від традиційного підходу з одним визначеним методом до програмно-визначеного підходу з використанням множини відповідних схем реалізації, моделей і методів;

2) вперше розроблено структурно-функціональну модель багаторівневої ієрархічної програмно-визначеної системи управління IT-інфраструктурою хмарного ЦОД через поєднання стратегічного і оперативного управління з автоматичним керуванням, а також надання програмно-визначених властивостей, яка дозволяє реалізувати задане управління ресурсами і навантаженням IT-інфраструктури з вибором стратегій, плануванням і управлінням їх реалізацією, автоматичним керуванням з урахуванням структури системи, ретроспективних даних її функціонування та дотриманням заданих показників якості угоди про рівень обслуговування;

3) вперше розроблено оригінальну інформаційну технологію управління IT-інфраструктурою хмарного ЦОД на основі запропонованої методології, моделей і методів управління яка відрізняється адаптивністю та використанням стратегічного управління з прогнозуванням за критеріями енергозбереження, забезпечення заданого рівня обслуговування, зниження операційних і капітальних витрат, що дозволяє підвищити ефективність використання ресурсів в умовах змінних навантажень;

4) вперше розроблено адаптивний метод комбінованого прогнозування навантаження на обчислювальні ресурси хмарного ЦОД з використанням усередненого, зваженого та оптимізаційного комбінування оцінок прогнозів, обчислених за альтернативними методами прогнозування та з адаптацією розміру навчальної вибірки, який відрізняється обчисленням та використанням певних типів комбінованих прогнозів (усередненого та зваженого) в реальному часі, отриманих за множиною альтернативних методів, що дає можливість оцінювати високоякісне прогнозоване значення для відомих видів змішаних навантажень в IT-інфраструктурі;

5) вперше розроблено метод інтегрованого управління ресурсами ІТ-інфраструктури хмарного ЦОД на основі динамічної моделі його станів із застосуванням стохастичного пошуку для виконання міграцій віртуальних машин, вивільнення, увімкнення і вимкнення фізичних серверів, а також прогнозування для адаптації до змін навантаження з метою досягнення сталого режиму роботи згідно визначених критеріїв, що дозволяє ефективніше управляти ресурсами ІТ-інфраструктури хмарного ЦОД із дотриманням заданих показників якості угоди про рівень обслуговування;

6) вперше розроблено метод управління розподіленим сховищем хмарного ЦОД на основі моделі дворівневого сховища з реплікацією та урахуванням кешування за розміром файлу і за кількістю транзакцій доступу до файлу для управління міграцією і з урахуванням кількості транзакцій доступу до блоків даних, об’єму вільного місця та затримки передачі даних між вузлами зберігання даних для управління реплікацією, який відрізняється застосуванням принципу гіперконвергентності, що забезпечило підвищення рівня надійності збереження даних, відмовостійкості та продуктивності їх оброблення в сучасних апаратно-програмних комплексах ЦОД;

7) вперше розроблено метод управління потужністю хмарного ЦОД на основі динамічної моделі його станів, який використовує запропоновані метрики оцінки стану хмарного ЦОД (коефіцієнт життєздатності віртуальної машини, індикатор дисбалансу фізичного сервера, коефіцієнт відношення необхідних ресурсів до середнього об’єму наявних ресурсів, поріг вільних ресурсів та метрика ємності ЦОД), враховує гетерогенність фізичних серверів, їх тип і кількість для обслуговування прогнозованого навантаження, який на відміну від існуючих підвищує якість обслуговування користувачів і зменшує споживання електроенергії, що дає можливість автоматично забезпечити ресурс потрібного типу, а також мінімальну затримку розгортання сервісів у хмарі;

8) отримав подальший розвиток декомпозиційно-компенсаційний підхід через надання адаптивності, багаторівневості та програмно-визначених властивостей за рахунок реалізації ефективного управління ресурсами ІТ-

інфраструктури хмарного ЦОД з переходом від вибору стратегій до планування і управління їх втіленням з урахуванням структури системи та її параметрів;

9) запропоновано удосконалення алгоритмів і методів стохастичного пошуку через розроблення нових методів управління ресурсами ІТ-інфраструктури, а саме методу рівномірної консолідації віртуальних машин з використанням ідеї імітації відпалу, двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку, методу динамічної консолідації і розміщення віртуальних машин на основі алгоритму навчання з підкріпленням, що відрізняються врахуванням процедур визначення станів системи на основі нових метрик, врахуванням достатньої кількості видів ресурсів і дискретності вимірів, що дозволяє ефективно визначати схему реалізації стратегії управління для різних режимів роботи хмарного ЦОД;

10) уточнено модель управління ІТ-інфраструктурою з координатором через застосування програмно-визначеного керування підсистемами, де координатор генерує керуючі впливи для програмно-визначених контролерів мережі, сховища і гіпервізорів, погоджуючи їх роботу з визначеною стратегією управління хмарним ЦОД, що дозволило застосовувати її в середовищах з програмно-визначеним функціонуванням.

Практична цінність інформаційної технології, а також розроблених алгоритмів, методів і підходів полягає у створенні методологічної бази розроблення і реалізації систем управління ІТ-інфраструктурою хмарних ЦОД і підвищення ефективності їх функціонування з подальшим їх застосуванням для розроблення підсистем, компонентів та інших складових систем управління ІТ-інфраструктурою провайдерів хмарних послуг.

До числа результатів, які мають найбільше практичне значення, належать: методологія управління на основі операторної форми вибору стратегії управління і схеми її реалізації; підхід до управління багаторівневою ієрархічною системою ресурсів ІТ-інфраструктури хмарного ЦОД; методи прогнозування навантаження хмарного ЦОД; методи управління ресурсами, навантаженням і потужністю хмарного ЦОД з урахуванням прогнозів; методи

управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД.

Ключові слова: центр оброблення даних, хмарні обчислення, система управління, прогнозування, стохастичний пошук, оптимізація, дворівневе сховище з реплікацією, інтернет речей.

ABSTRACT

Zharikov E. V. Information technology for the cloud data center IT infrastructure management. – Qualifying scientific work as a manuscript.

Thesis for scientific degree of Doctor of Technical Sciences on the specialty 05.13.06 – information technologies. – National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ministry of Education and Science of Ukraine, Kyiv, 2020.

The dissertation solves the scientific and practical problem of effective management of the IT infrastructure of cloud data centers under conditions of uncertainty and variable workloads by creating a management methodology and on its basis the development and application of management information technology to provide cloud services with specified quality indicators to the end user. The proposed information technology is developed ensuring the efficient functioning of the IT infrastructure of the cloud service provider's data center by increasing the resource usage efficiency under conditions of variable workload, which unlike the known methods takes into account the developed methodology, essential characteristics of cloud computing, heterogeneity of IT infrastructure, its multilevel and hierarchy and uses adaptation to unpredictable and mixed workloads at the expense of forecasting, which allowed to ensure SLA requirements while reducing operating and capital costs.

The developed information technology is based on the concept of management of IT infrastructure of a cloud data center, which combines the following developed findings, approaches, and methods: the operator form of setting, analyzing and solving problems of IT infrastructure of the cloud data center; identifying and implementing dedicated resource and workload management strategies, as well as their implementation schemes; decomposition of cloud IT infrastructure at three levels (infrastructure, platforms, and applications); taking into account traditional, convergent

and hyperconverged architectures of modern cloud data centers; application of the adaptive method of combined workload forecasting to determine the control influences on the IT infrastructure of cloud data centers; developing the Method of Integrated Resource Management for heterogeneous data centers based on SLA violations, power consumption and required power at the next management step; application of stochastic methods (beam search, simulated annealing, reinforcement learning) for implementation of particular strategies for IT infrastructure management of cloud data centers; application of a distributed two-level storage management method for hyperconverged systems; taking into account new metrics of the IT infrastructure state namely instantaneous and average viability coefficients of virtual machine, physical server imbalance indicator, the ratio of necessary resources to the average available resources, threshold of available resources) to determine the current management strategy; accounting of software-defined controllers for management of three primary data center resources namely computing, storage, network; the combination of centralized management using global data center manager and the decentralized management on the physical server level, depending on the chosen management strategy at the current management step.

To solve the problem of effective management of the IT infrastructure of the cloud data center and solving the correspondent set of tasks, the following were used: systems theory, methods of hierarchical systems theory, methods of mathematical programming, methods of operations research and decision theory, methods of mathematical and simulation modeling, methods of artificial intelligence theory, stochastic and heuristic search methods, forecasting methods, mathematical statistics methods, and cloud service models. The reliability and validity of the obtained results are conditioned by the correct use of the mathematical apparatus and are confirmed by the results of computational experiments.

The scientific novelty of the obtained results is determined by the following theoretical and practical results obtained by the author:

1) for the first time a methodology for managing the IT infrastructure of a cloud data center has been developed. It is based on the operator formulation, analysis and solution of management problems under uncertainty and variable loads, which

differs the choice of management strategies in different modes of operation of the cloud data center by defining appropriate models by combining criteria and constraints , as well as by identifying appropriate implementation schemes that allow one to move from a traditional approach with one defined method to programs when managing a cloud datacenter with software-defined approach using a plurality of respective schemes implementation models and methods;

2) for the first time a structural and functional model of a multilevel hierarchical software-defined system for managing the IT infrastructure of a cloud data center has been developed by combining strategic and operational management with automatic control, as well as providing software-defined properties, which allows to realize the given management of resources and load of IT infrastructures with the choice of IT infrastructure, planning and management of their implementation, automatic control taking into account the structure of the system, retrospective data of its functioning and compliance with SLA quality indicators;

3) for the first time an original information technology for managing the IT infrastructure of a cloud data center has been developed based on the proposed methodology, models and management methods, characterized by adaptability and the use of strategic management with energy saving criteria forecasting, providing a specified level of service, reducing operational and capital costs, which allows to increase resource efficiency in conditions of variable loads;

4) for the first time an adaptive method of combined workload forecasting for cloud data center computing resources has been developed using averaged, weighted and optimized combination of forecast estimates calculated by alternative forecasting methods and with the adaptation of training window size, which differs from the calculation and use of certain forecasts in real-time, obtained through a variety of alternative methods, which allows one to evaluate high-quality forecasts for known types of mixed workloads in IT infrastructure;

5) for the first time a method for integrated management of IT infrastructure of cloud data center has been developed based on a dynamic model of its states using stochastic search to perform virtual machine migration, release, switching on and switching off of physical servers, as well as forecasting to adapt to changes in workload

to achieve a sustainable mode defined criteria, which allows one to more efficiently manage cloud datacenter IT infrastructure while meeting the service level agreement ;

6) for the first time a distributed datacenter cloud management method has been developed based on a two-tier model with replication and file size caching and the number of file access transactions to manage migration and taking into account the number of data block access transactions, free space, and data transmission delays between the storage nodes to manage replication, characterized by the application of the principle of hyperconvergence, which provides an increase in the level of reliability of data storage, fault tolerance and efficiency of their processing in modern hardware and software of a complex data center;

7) a cloud datacenter power management method has been developed for the first time based on a dynamic model of its states, which uses the proposed metrics for cloud datacenter condition detection (virtual machine viability metric, physical server imbalance indicator, ratio of required resources to the average amount of available resources, and time threshold of metric resources capacities of the data center), taking into account the heterogeneity of physical servers, their type and quantity for maintenance of the predicted workload, which unlike the existing ones improves a customer service and reduces energy consumption, which makes it possible to automatically provide the desired resource type and minimal deployment delay of services in the cloud;

8) the decomposition-compensation approach has been further improved by providing adaptability, multilevel and software-defined properties through the implementation of efficient management of cloud datacenter IT infrastructure with the transition from strategy selection to planning and management of their implementation, taking into account the structure of the system and its parameters;

9) the algorithms and methods of stochastic local search were further improved by developing new methods of IT infrastructure resource management, namely the method of uniform consolidation of virtual machines using the simulating annealing algorithm, a two-stage method of managing the resources of a cloud data center based on the beam search algorithm and the method of dynamic machine consolidation based on the algorithm of reinforcement learning, which unlike the

existing ones differentiate the procedures for determining the state of the system based on new metrics, took into account a sufficient number of types of resources and discreteness of measurements that allows to determine effectively the scheme of implementation of management strategy for operation in different cloud data center modes;

10) an IT infrastructure management model with a coordinator has been further improved through the application of software-defined subsystem management, where the coordinator generates management impacts for software-defined network controllers, storages and hypervisors, coordinating their work with a defined cloud datacenter management strategy, which allowed it to be deployed with software-defined functionality.

Practical value of the information technology, as well as developed methods and approaches, is to create a methodical basis for the development and implementation of management systems for IT infrastructure of cloud data centers and increase their efficiency with their subsequent application for the development of subsystems, components and other parts of the IT infrastructure management systems for cloud service providers.

The most practical results include: management methodology based on the operator form of the management strategy choice and scheme of its implementation; an approach to managing a multi-level hierarchical cloud infrastructure data center IT infrastructure resources; methods of forecasting the workload in cloud data center; methods of managing the resources, workload and power of the cloud data center using forecasts; methods for managing replication and cross-level data migration in a cloud datacenter storage systems.

Keywords: data center, cloud computing, management system, forecasting, stochastic search, optimization, two-level storage with replication, internet of things.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1. Rolik O., Telenyk S., Zharikov E. IoT and Cloud Computing: The Architecture of Microcloud-Based IoT Infrastructure Management System / In P. Kocovic, R. Behringer, M. Ramachandran, & R. Mihajlovic (Eds.), Emerging Trends and

Applications of the Internet of Things. – Hershey, PA: IGI Global. 2017. – С. 198-234. ISBN10: 1522524371 doi:10.4018/978-1-5225-2437-3.ch008

2. Telenyk S., Zharikov E., Rolik O. Consolidation of Virtual Machines Using Stochastic Local Search // Advances in Intelligent Systems and Computing. – Springer, Cham, 2017. – Vol. 689. – С. 523-537. (Scopus).

3. Telenyk S., Zharikov E., Rolik O. Architecture and conceptual bases of cloud IT infrastructure management // Advances in Intelligent Systems and Computing. – Springer, Cham, 2017. – Vol. 512. – С. 41-62. (Scopus).

4. Rolik O., Telenyk S., Zharikov E. Management of services of a hyperconverged infrastructure using the coordinator // Advances in Intelligent Systems and Computing. – Springer, Cham, 2018. – Vol. 754. – С. 456-467. (Scopus).

5. Zharikov E., Telenyk S., Rolik O. Method of Distributed Two-Level Storage System Management in a Data Center // Advances in Intelligent Systems and Computing. – Springer, Cham, 2019. – Vol. 938. – С. 301-315. (Scopus).

6. Zharikov E. Topical questions of implementation of information services in a network of university // TEKA Kom. Mot. i Energ. Roln. – 2010, Vol 10B. – С. 331-337. (Bibliografia Geografii Polskiej, EBSCO, Index Copernicus).

7. Zharikov E. Analysis of the theoretical and applied aspects of modern IT infrastructure // Teka. Commission of motorization and energetics in agriculture. – 2013, Vol. 13, №4. – С. 297-306. (Bibliografia Geografii Polskiej, EBSCO, Index Copernicus).

8. Samozdra M., Zharikov E., Samozdra O. Implementation of automated informational interactions as a part of integrated information-processing system // Teka. Commission of motorization and energetics in agriculture. – 2014, Vol. 14, №1. – С. 229-237. (Bibliografia Geografii Polskiej, EBSCO, Index Copernicus).

9. Rolik O., Telenyk S., Zharikov E. Service quality management in microcloud-based IoT infrastructure //Czasopismo Techniczne. – 2016. – Vol. 2-E(12). – С. 231-244. DOI: 14467/2353737XCT.16.243.6042 (BazTech, Bibliografia Geografii Polskiej, EBSCO, Index Copernicus).

10. Telenyk S., Rolik O., Zharikov O., Serdiuk Y. Energy efficient data center resources management using beam search algorithm //Czasopismo Techniczne. – 2018. – Т. 2018. – №. 4. – С. 127-138. (BazTech, Bibliografia Geografii Polskiej, EBSCO, Index Copernicus).

11. Жариков Э. В., Овчинников В. Л. Система передачи мультимедийных данных в локальных сетях // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2008. – №9. Частина 1. – С. 142-146.

12. Зубов Д. А., Ульшин В. А., Жариков Э. В., Григоренко М. С. Концепция программно-аппаратного комплекса долгосрочного прогнозирования средней температуры воздуха на базе статистико-гидродинамических моделей // Збірник наукових праць Українського державного геологорозвідувального інституту. 2007. – №4. – С. 223-226.

13. Жаріков Е.В., Солдатенко В.Ю. Методика порівняльного аналізу бездротових технологій // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2011. – № 11. Частина 2. – С. 250-253.

14. Жариков Э.В., Шмитько А.А. Автоматизация подготовки научных изданий к печати // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2006. – №1(95). – С. 237-240.

15. Жариков Э.В. Публикация и использование знаний в глобальной сети // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2003. – №4(62). – С. 36-40.

16. Жариков Э.В. Разработка методов представления и организации знаний в распределенной экспертной системе. // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2005. – №3(85). – С. 89-95.

17. Жариков Э. В. Основные направления оптимизации ИТ-инфраструктуры учебных заведений // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2011. – № 3. – С. 66–71.

18. Теленик С.Ф., Ролик А.И., Жариков Э.В. Управление распределением виртуальных машин в ЦОД // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+. – 2016. – № 64. – С. 90–99.

19. Жаріков Е.В., Сердюк Е.А. Метод консолидации виртуальных машин на основе лучевого поиска / Е.В.Жаріков // Автомобіль і електроніка. Сучасні технології. – 2017. – No12. – С. 180–186.

20. Ролік О.І., Теленик С.Ф., Жаріков Е.В. Управління рівнем послуг в системі інтернету речей з мікрохмарною архітектурою // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+. – 2017. – № 65. – С. 110–117.

21. Жаріков Е.В. Керування ресурсами хмарних центрів обробки даних на основі евристичного пошуку // Проблеми програмування. – 2017. – № 4. – С. 16-27.

22. Жаріков Е. В. Динамічне розміщення віртуальних машин на основі навчання з підкріпленням в хмарних центрах обробки даних / Е. В. Жаріков, А. А. Коваль, Р. А. Терентьєв. // Наукові вісті Далівського університету. - 2017. - № 13. - Режим доступу: http://nbuv.gov.ua/UJRN/Nvdu_2017_13_4

23. Жаріков Е. В., Моделювання динаміки хмарного центру обробки даних у просторі станів // Моделювання та інформаційні технології. – 2018. – № 84(6). – С. 125-134

24. Жаріков Е.В. Інтегроване управління ресурсами хмарного центру обробки даних на основі віртуальних машин // Матем. машини і системи.– 2018. – № 2. – С. 21-32.

25. Жаріков Е. В., Структурна оптимізація моделей прогнозу споживання обчислювальних ресурсів в умовах віртуалізації // Електронне моделювання. – 2018. - №5. - С. 49-66.

26. Жаріков Е. В. Метод управління дворівневим сховищем віртуалізованого центру обробки даних // Проблеми програмування. – 2018. – № 4. – С. 3-14.

27. Telenyk S., Nowakowski G., Zharikov E., Vovk Y. Information technology for web-applications design and implementation // Адаптивні системи автоматичного управління, К: Політехніка. – 2019. – Т.1, №34. – С. 138-151. (Google Scholar, WorldCat).

28. Telenyk S., Zharikov E. Operator form to formulate, analyze and solve the cloud data center IT infrastructure management tasks // Адаптивні системи автоматичного управління, К: Політехніка. – 2019. – Т.2, №35. – С. 25-39. (Google Scholar, WorldCat).

Статті в збірниках матеріалів конференцій, які індексуються Scopus та IEEE

29. Telenyk S. An approach to Software Defined Cloud Infrastructure management / S. Telenyk, E. Zharikov, O. Rolik // Proc. of the XI International Scientific and Technical Conference “Computer Science and Information Technologies Congress on Information Technology” (CSIT 2016) 6–10 September, Lviv, Ukraine. – 2016 p. – p. 21–26.

30. Telenyk S., Zharikov E., Rolik O., “An approach to virtual machine placement in cloud data centers,” In proc. of the 2016 International Conference Radio

Electronics & Info Communications (UkrMiCo) 11–16 September, Kyiv, Ukraine. – 2016 p. – p. 1–6.

31. Rolik O., Zharikov E., Telenyk S. Microcloud-based architecture of management system for IoT infrastructures //Problems of Infocommunications Science and Technology (PIC S&T), Third International Scientific-Practical Conference. – IEEE, 2016. – pp. 149-151.

32. Rolik, O., Telenyk, S., Zharikov, E., & Yasochka, M. (2016, December). Decomposition-compensation approach to microcloud-based IoT infrastructure management. In Proc. of 3rd World Forum on Internet of Things (WF-IoT) 2016 (pp. 603-608). IEEE. DOI: 11109/WF-IoT.2016.7845413

33. Rolik O., Telenyk S., Zharikov E., Samotyy V., Dynamic Virtual Machine Allocation Based on Adaptive Genetic Algorithm // The Eighth International Conference on Cloud Computing, GRIDs, and Virtualization. – 2017. – pp. 108-114.

34. Rolik O., Zharikov E., Kolesnik V., Yasochka M., “Rule-based Algorithmic Approach for Solving Problems of Impact Analysis in Access Networks,” In proc. of the 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON) May 29 – June 2, Kyiv, Ukraine. – 2017 – p. 1161–1166.

35. Telenyk S., Zharikov E., and Rolik O. “Consolidation of Virtual Machines Using Simulated Annealing Algorithm” in Proc. of the XIIth International Scientific and Technical Conference “Computer Science and Information Technologies” (CSIT 2017) 5–8 September, Lviv, Ukraine. – 2017. – pp. 117–121.

36. Telenyk S., Bidyuk P., Zharikov E. and Yasochka M., Assessment of cloud service provider quality metrics, 2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo), Odesa, 2017, pp. 1-5.

37. Telenyk S., Zharikov E., Rolik O., "An Integrated Approach to Cloud Data Center Resource Management" //Problems of Infocommunications Science and Technology (PIC S&T), 4th International Scientific-Practical Conference. – IEEE, 2017. – pp. 211-218. DOI: 11109/INFOCOMMST.2017.8246382

38. Telenyk S., Rolik O., Zharikov O., Serdiuk Y. “Consolidating Virtual Machines with the Use of Beam Search Algorithm”, The Fourth International Conference on Automatic Control and Information Technology (ICACIT’17), 2017, pp. 34-46.

39. Rolik O., Zharikov E., Koval A. and Telenyk S., "Dynamic management of data center resources using reinforcement learning," 2018 14th International

Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 2018, pp. 237-244. doi: 11109/TCSET.2018.8336194"

40. Zharikov E., Rolik O., Telenyk S. A Decomposition Approach to Hierarchical Management of Cloud Data Center Services //2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). – IEEE, 2018. – Т. 1. – С. 42-47.

41. Rolik O., Zharikov E., Yasochka M., Butenko M. The Method of Impact Analysis for Access Networks with RIP and OSPF Protocols // 2018 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo), Odesa, Ukraine, 2018, pp. 1-7.

42. Telenyk S., Zharikov E. and Rolik O., "Modeling of the Data Center Resource Management Using Reinforcement Learning," 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2018, pp. 289-296. doi: 11109/INFOCOMMST.2018.8632064

Статті і тези доповідей в збірниках матеріалів конференцій

43. Жаріков Е. В. Технології віртуалізації на базі VIRTUAL SERVER в навчальному процесі // Збірник наукових праць Східноукраїнського національного університету імені Володимира Даля, Міжнародні Далівські читання XV Нуково-практична конференція "Університет і регіон:проблеми сучасної освіти"/ За заг.ред.проф. Голубенка О.Л.– Луганськ: вид-во Східноукраїнського національного університету імені Володимира Даля, 2009. – С. 247-249.

44. Жариков Э. В. Актуальные вопросы оптимизации IT-инфраструктуры учебных заведений // Комп’ютерні науки для інформаційного суспільства: Матеріали міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 22-23 грудня 2010 р.). – Луганськ: Вид-во «Ноулідж», – С. 98-102.

45. Солдатенко В. Ю., Жариков Э. В. Методика сравнительного анализа беспроводных технологий для проектирования сетей передачи данных // Комп’ютерні науки для інформаційного суспільства: Матеріали міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 22-23 грудня 2010 р.). – Луганськ: Вид-во «Ноулідж», 201 – С. 126-128.

46. Кордубайло С. А., Жариков Э. В. Планирование внедрения технологий виртуализации на предприятии // Комп’ютерні науки для інформаційного

суспільства: Матеріали ІІ Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 23-24 листопада 2011 р.). – Луганськ: Вид-во «Ноулідж», 2011. – С. 109-112.

47. Альсаясні М., Жаріков Е. В. Підвищення якості роботи інформаційних сервісів корпоративної мережі // Комп'ютерні науки для інформаційного суспільства: Матеріали ІІ Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 23-24 листопада 2011 р.). – Луганськ: Вид-во «Ноулідж», 2011. – С. 82-85.

48. Грищенко Е. В., Жариков Э. В. Уровневое разбиение элементов ИТ-инфраструктуры // Комп'ютерні науки для інформаційного суспільства: Матеріали ІІІ Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 12-13 грудня 2012 р.). – Луганськ: Вид-во «Ноулідж», 2012. – С. 107-108.

49. Жариков Э. В., Альсаясни М., Лукутин О. В. Анализ систем мониторинга ИТ-инфраструктуры и разработка подхода к их интеграции // Комп'ютерні науки для інформаційного суспільства: Матеріали ІІІ Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 12-13 грудня 2012 р.). – Луганськ: Вид-во «Ноулідж», 2012. – С. 128-132.

50. Удовенко Д. В., Жариков Э. В. Облачные вычисления как инновационное направление информационно-коммуникационных технологий // Комп'ютерні науки для інформаційного суспільства: Матеріали ІІІ Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 12-13 грудня 2012 р.). – Луганськ: Вид-во «Ноулідж», 2012. – С. 208-21.

51. Жариков Э. В., А. Салем Повышение качества информации в современной ИТ-инфраструктуре // Информатика и вычислительная техника: сборник научных трудов 5-й Всероссийской научно-технической конференции аспирантов, студентов и молодых ученых ИВТ-2013 / под ред. Н. Н. Войта. – Ульяновск: УлГТУ, 2013. –С. 69-74.

52. Жариков Э. В., Альсаясни М. Об одном подходе к интеграции систем мониторинга в современной ИТ-инфраструктуре // Информатика и вычислительная техника: сборник научных трудов 5-й Всероссийской научно-технической конференции аспирантов, студентов и молодых ученых ИВТ-2013 / под ред. Н. Н. Войта. –Ульяновск: УлГТУ, 2013. –С. 65-69.

53. Жариков Э. В. Критерии оптимизации ИТ-инфраструктуры предприятия // Информационно-компьютерные технологии в экономике,

образовании и социальной сфере. Выпуск 8. – Симферополь : ФЛП Бондаренко О.А., 2013. С. 6-7

54. Obinna J., Zharikov E. Architectural Development of Pivate Cloud for Mid Business Enterprises // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 42-46.

55. Adetola R., Zharikov E. Challenges in cloud computing // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 33-35.

56. Жариков Э. В., Алдееб С. Неструктурированная информация предприятия, качественный и количественный аспект // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 52-56.

57. Жариков Э. В., Альсаясни М., Лукутин О. В. Идентификация инцидента методом распознавания образов в системах мониторинга // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 56-58.

58. Жариков Э. В., Алдееб С. Моделирование ИТ-инфраструктуры с использованием Archimate // Комп'ютерні науки для інформаційного суспільства: Матеріали V Міжнародної науково-практичної конференції аспірантів та молодих вчених (м. Сєвєродонецьк, 23 грудня 2014 р.). – Сєвєродонецьк: Вид-во СНУ ім.В.Даля, 2014. – С. 21-26.

59. Жариков Э. В., Алдееб С. Анализ компонентов ИТ-инфраструктуры в контексте обработки информации предприятия // Комп'ютерні інтелектуальні системи та мережі: Матеріали VIII Всеукраїнської науково практичної WEB конференції аспірантів, студентів та молодих вчених (24-26 березня 2015 р.). – Кривий Ріг: ДВНЗ «Криворізький національний університет», 2015. – С. 23-26.

60. Жариков Э. В. Архитектура системы управления и мониторинга производительности программно-определяемой ИТ-инфраструктуры // «АВІА-2015»: Матеріали XII Міжнародної науково-технічної конференції (28-29 квітня 2015 р.). – Київ: Національний авіаційний університет, 2015. – С. 638-641.

61. Жариков Э. В. Гиперконвергентная архитектура, анализ возможностей применения в ЦОД // Комп’ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції аспірантів, студентів та молодих вчених (22-24 березня 2016 р.). – Кривий Ріг: ДВНЗ «Кривор. нац. універ.», 2016. – С. 14-19.

62. Коваль А., Жаріков Е. Порівняльний аналіз існуючих методів управління ресурсами в умовах хмарних обчислень // Комп’ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції КІСМ-2017 (22-24 березня 2017 р.). – Кривий Ріг: ДВНЗ «Кривор. нац. універ.», 2017. – С. 8-10.

63. Сягайло Т., Жаріков Е. Порівняльний аналіз алгоритмів для управління розміщенням віртуальних машин // Комп’ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції КІСМ-2017 (22-24 березня 2017 р.). – Кривий Ріг: ДВНЗ «Кривор. нац. універ.», 2017. – С. 10-13.

64. Терентьєв Р., Жаріков Е. Порівняльний аналіз засобів моделювання інфраструктури хмарних обчислень // Комп’ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції КІСМ-2017 (22-24 березня 2017 р.). – Кривий Ріг: ДВНЗ «Кривор. нац. універ.», 2017. – С. 13-16.

65. Andrii Koval, Roman Terentiev, Eduard Zharikov, Comparative Analysis of Modeling Methods of Infrastructure of Cloud Computing, Science and Technology of the XXI Century: the XVIII All-Ukrainian Students R&D Conference, (Kyiv, December 07, 2017) / NTUU „Igor Sikorsky Kyiv Polytechnic Institute“. – Part IV. – Kyiv, 2017. – P.124.

66. Terentiev R., Koval A., Zharikov E., Comparative Analysis of Resource Management Methods in Cloud Computing, Science and Technology of the XXI Century: the XVIII All-Ukrainian Students R&D Conference, (Kyiv, December 07, 2017) / NTUU „Igor Sikorsky Kyiv Polytechnic Institute“. – Part V. – Kyiv, 2017. – P.125.

67. Сягайло Т.А. Способи налаштування сховищ в гіперконвергентних системах./ Жаріков Е.В., Сягайло Т.А., Коваль А.А. // Актуальні наукові дослідження в сучасному світі. - 2018. Вып. 4(36), ч. 3 - С.41 - 51.

68. Жаріков Е.В., Сердюк Є.О. Консолідація віртуальних машин з використанням променевого пошуку // Інформатика та обчислювальна техніка ІОТ-2017. – Київ: НТУУ "КПІ ім. І. Сікорського", 2017. – С. 79-84

69. Коваль А.А., Жаріков Е.В. Метод розміщення віртуальних машин на основі навчання з підкріпленням // Інформатика та обчислювальна техніка: Матеріали наукової конференції студентів, магістрантів та аспірантів ІОТ-2018 (23 – 24 квітня 2018). – Київ: НТУУ "КПІ ім. І. Сікорського", 2018. – С. 29-35.

70. Сягайло Т.А., Жаріков Е.В., Управління процесом збереження даних в гіперконвергентних системах // Інформатика та обчислювальна техніка: Матеріали наукової конференції студентів, магістрантів та аспірантів ІОТ-2018 (23 – 24 квітня 2018). – Київ: НТУУ "КПІ ім. І. Сікорського", 2018. – С. 119-122.

71. Терентьєв Р.А., Жаріков Е.В., Прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень // Інформатика та обчислювальна техніка: Матеріали наукової конференції студентів, магістрантів та аспірантів ІОТ-2018 (23 – 24 квітня 2018). – Київ: НТУУ "КПІ ім. І. Сікорського", 2018. – С. 123-126.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

DAS	Direct Attached Storage
EWMA	Exponentially weighted moving average
IaaS	Infrastructure as a service
IoT	Internet of things
MAPE	Mean absolute percentage error
MPC	Model predictive control
NAS	Network Attached Storage
PaaS	Platform as a service
QoS	Quality of service
SaaS	Software as a service
SAN	Storage Area Network
SDDC	Software Defined Data Center
SDN	Software Defined Networks
SLA	Угода про рівень обслуговування (Service Level Agreement)
VDI	Virtual desktop infrastructure
АПВМ	Адаптивний програмно-визначений менеджер
ВМ	Віртуальна машина
ГА	Генетичний алгоритм
ГІ	Гіперконвергентна інфраструктура
ІКТ	Інформаційно-комунікаційні технології
ІУР	Інтегроване управління ресурсами
МВМ	Монітор віртуальної машини
МФС	Менеджер фізичного сервера
НП	Навчання з підкріпленням
ПЗ	Програмне забезпечення
САПП	Середня абсолютна похибка в процентах
СЗД	Система збереження даних
СУІ	Система управління ІТ-інфраструктурою
ФС	Фізичний сервер
ЦОД	Центр оброблення даних
ЦП	Центральний процесор

ВСТУП

Дисертаційна робота присвячена вирішенню науково-практичної проблеми забезпечення ефективного функціонування ІТ-інфраструктури хмарних центрів оброблення даних у спосіб управління обчислювальними ресурсами з метою надання хмарних послуг із заданими показниками якості кінцевому користувачеві.

Дослідження спрямоване на розроблення нових, подальший розвиток та вдосконалення існуючих моделей, методів та засобів управління апаратно-програмними комплексами центрів оброблення даних, що надають інформаційні послуги, пов’язані з обробкою, передачею і зберіганням даних в умовах віртуалізації, програмно-визначених середовищ і хмарних обчислень.

Актуальність теми. Цифрова трансформація суттєво впливає на всі сфери життєдіяльності людини і вимагає вирішення низки проблем організації обчислень, оброблення, передачі і зберігання даних, що потребує розроблення і впровадження нових підходів, технологій і методів управління ІТ-інфраструктурою. ІТ-інфраструктура забезпечує реалізацію інформаційних процесів і лежить в основі функціонування бізнес-процесів компанії. Від ефективного функціонування ІТ-інфраструктури напряму залежить реалізація місії компанії і її конкурентоспроможність.

Існує багато конфігурацій і схем реалізації ІТ-інфраструктури, але основне навантаження, що виникає при наданні сучасних ІТ-послуг, обслуговується ІТ-інфраструктурою центрів оброблення даних (ЦОД). При цьому у процесі еволюції ІТ-інфраструктури компанії виникають проблеми технічного і організаційного характеру, рішення яких залежить від багатьох факторів, в тому числі від рівня розвитку архітектури, сучасних тенденцій в області віртуалізації і програмно-визначених систем, хмарних обчислень і аутсорсингу.

Дослідження і розробки в галузі управління ІТ-інфраструктурою ЦОД проводяться міжнародними організаціями (Distributed Management Task Force, The Open Group, Cloud Native Computing Foundation, Storage Networking Industry Association, Uptime Institute, IEEE та ін.), аналітичними агенціями (Gartner, IDC, Forrester Research, RightScale), великими корпораціями (Microsoft, Google,

Amazon, IBM, Hewlett-Packard, Cisco та ін.). Управлінню IT-інфраструктурою присвячено ряд праць (Глушков В.М., Палагін О.В. та ін.). Управління IT-інфраструктурою спирається на такі напрями наукових досліджень як: системний аналіз (Берталанфі Л. фон, Вінер М., Вернадський В.І., Клір Дж., Згуровський М.З. та ін.), математичні методи вирішення задач оптимізації (Нейман Д. фон, Данціг Д.Б., Канторович Л.В. та ін.), інформаційні технології та системи (Гроувер Д., Сатер Р., Баррозо Л. та ін.), управління якістю послуг (Демінг У., Джуран Д., Ісікава К., Кросбі Ф. та ін.), теорія ієрархій (Петті Г., Уайт Л., Аллен Т. та ін.), теорія управління багато-об'єктними багаторівневими системами (Кунцевич В.М., Лебедев Д.В., Месарович М., Беллман Р., Воронов Є.М. та ін.), теорія штучного інтелекту (Хопфілд Д.Д., Голдберг Д. та ін.).

Але наведені дослідження не вирішують всі проблеми, пов'язані з управлінням IT-інфраструктурою хмарного ЦОД провайдерів хмарних послуг. Недостатньо опрацьовані аспекти управління, пов'язані з управлінням ресурсами з дотриманням заданих вимог угоди про рівень обслуговування (SLA) в умовах невизначеності, змін режимів роботи і навантажень, раціонального використання ресурсів IT-інфраструктури хмарного ЦОД з урахуванням різних стратегій і критеріїв, зменшення операційних витрат на підтримку ефективної роботи IT-інфраструктури, вибір стратегій, моделей і критеріїв управління в умовах роботи програмно-визначених систем і гібридних IT-інфраструктур різних поколінь та ін.

Останнім часом відокремились два основних варіанти реалізації інформаційних процесів компаніями та організаціями: з використанням приватної (власної) IT-інфраструктури і з використанням IT-інфраструктура як послуги. Більш широке розповсюдження отримала IT-інфраструктура як послуга, що обумовлено високими початковими витратами на побудову власної IT-інфраструктури та високими операційними витратами при її експлуатації. Сучасні провайдери ЦОД адаптуються до появи нових вимог і навантажень через поступове зменшення кількості традиційних та багатопрофільних IT-інфраструктур і впровадження стандартизованих і взаємозамінних рішень, що базуються на програмно-визначеному підході до управління IT-

інфраструктурою ЦОД. При цьому існуючі рішення і підходи до управління ІТ-інфраструктурою ЦОД в умовах віртуалізації серверів, мереж і сховищ даних в недостатній мірі відповідають сучасним потребам і викликам бізнесу, що обумовлено високою динамікою розвитку інформаційних процесів в цифровому суспільстві, появою новітніх парадигм та підходів до реалізації інформаційних процесів, необхідністю існування і функціонування гібридних ІТ-інфраструктур різних поколінь.

Таким чином, сформувався перелік завдань, ефективне розв'язання яких дозволяє істотно покращити показники управління сучасною ІТ-інфраструктурою у спосіб розроблення підходів, технологій і методів управління програмно-апаратними ресурсами та інформаційними процесами. Серед показників управління ІТ-інфраструктурою необхідно виділити такі: зменшення споживання електроенергії, підвищення якості обслуговування споживачів, зменшення капітальних та операційних витрат, зменшення кількості збоїв та простоїв. Також, такі важливі характеристики сучасної ІТ-інфраструктури, як багаторівневість і ієрархічність треба врахувати через інтеграцію впливів управління, що виробляються на верхніх рівнях системи з впливами керування на нижньому рівні системи.

Крім того, набула поширення концепція хмарних обчислень, яка реалізує обчислення на замовлення (англ. utility computing) у спосіб використання хмарних ЦОД з метою забезпечення споживачів високоякісними і високопродуктивними ІТ-послугами. Хмарні ЦОД є найбільш динамічними, складними системами з централізованим управлінням, що забезпечують надання тарифікованого доступу до інформаційних платформ, процесів, застосунків та обчислювальних ресурсів на основі сервісних моделей хмарних послуг. Складність хмарних ЦОД вимагає передових інформаційних технологій з управління ІТ-інфраструктурою для забезпечення заданої якості обслуговування і забезпечення гарантованої продуктивності. Забезпечення заданої якості надання хмарних послуг в умовах змінних навантажень є одним з основних завдань при управлінні ІТ-інфраструктурою хмарного ЦОД. Правильно визначені технології, підходи і методи управління ІТ-інфраструктурою

дозволять досягти відповідності вимогам угоди про рівень обслуговування (SLA), заданої ефективності роботи сервісів, використання ресурсів, енергозбереження та збільшення прибутку хмарних провайдерів. Отже, виникає науково-практична проблема забезпечення ефективного функціонування IT-інфраструктури хмарних ЦОД через створення методології та на її основі математичних моделей, методів та інформаційної технології управління з метою надання хмарних послуг із заданими показниками якості кінцевому користувачеві.

Таким чином, виключна актуальність питань теоретико-методологічного обґрунтування впровадження підходів і методів управління IT-інфраструктурою хмарних ЦОД і розроблення інформаційної технології, що реалізує зазначені методи і підходи, зумовили і визначили мету, завдання та зміст дисертації.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконана у відповідності до плану наукових досліджень в рамках науково-дослідних робіт: "Хмарна платформа розроблення і управління функціонуванням критичних IT-інфраструктур, що опрацьовують великі обсяги даних", 2017-2019 рр., номер державної реєстрації 0117U000537; "Розробка та впровадження системи управління IT-інфраструктурою з консолідованими інформаційно-обчислювальними ресурсами", 2014-2016 рр., номер державної реєстрації 0115U000322; "Розроблення і дослідження моделей, методів та технологій проектування, програмування і управління хмарними IT-інфраструктурами", 2013-2015 рр., номер державної реєстрації 0113U002285; "Платформа розроблення, експлуатації і розвитку критичних IT-інфраструктур для роботи з великими даними", 2016-2018 рр., номер державної реєстрації 0116U003801.

Мета і завдання дослідження. Метою роботи є створення методології та на її основі математичних моделей, методів та інформаційної технології як системного підходу до вирішення проблеми ефективного управління IT-інфраструктурою хмарного ЦОД в умовах невизначеності та змінних навантажень, що дозволить забезпечити заданий рівень обслуговування користувачів та зниження операційних і капітальних витрат.

Для досягнення зазначеної мети необхідно обґрунтувати і розробити теоретичні положення, методологічні підходи і науково-практичні рекомендації щодо розвитку теорії автоматизованого управління ІТ-інфраструктурою хмарного ЦОД на основі системного підходу до оброблення інформації, прогнозування і оптимізації.

Досягнення поставленої мети передбачає розв’язання таких теоретичних, методологічних і практичних задач:

1) аналіз існуючих підходів, технологій, моделей і методів управління ІТ-інфраструктурою ЦОД провайдерів хмарних послуг, уточнення проблеми і задач дослідження;
2) розроблення методології управління ІТ-інфраструктурою хмарного ЦОД як систему новітніх підходів, моделей і методів з використанням стратегій управління і схем їх реалізації для відповідних умов функціонування;
3) розроблення моделей і методів прогнозування змішаних навантажень на ресурси ІТ-інфраструктури хмарного ЦОД в умовах невизначеності з використанням принципів адаптації та комбінування;
4) розроблення комплексу моделей і методів інтегрованого управління ресурсами, потужністю і сховищем ІТ-інфраструктури хмарного ЦОД на базі алгоритмів і методів стохастичного локального пошуку з використанням прогнозування змінних навантажень, нових метрик оцінювання стану та дотриманням вимог угоди про рівень обслуговування;
5) подальший розвиток алгоритмів і методів стохастичного локального пошуку з урахуванням особливостей управління ресурсами і навантаженням ІТ-інфраструктури хмарного ЦОД в умовах невизначеності;
6) розроблення архітектури багаторівневої програмно-визначеної системи управління ІТ-інфраструктурою хмарного ЦОД і концепції інформаційної технології на базі підходів і методів інтегрованого та ієрархічного управління;
7) подальший розвиток декомпозиційно-компенсаційного підходу до управління хмарним ЦОД з урахуванням адаптивності, багаторівневості та програмно-визначених підходів його функціонування;

8) розроблення і реалізація інформаційної технології управління ІТ-інфраструктурою хмарного ЦОД та експериментальне дослідження її ефективності, а також розроблених моделей, алгоритмів і методів

Об'єктом дослідження є процеси управління ІТ-інфраструктурою центру оброблення даних провайдера хмарних послуг.

Предметом дослідження є методологія, моделі і методи процесів управління ІТ-інфраструктурою центру оброблення даних провайдера хмарних послуг.

Методи дослідження. Для вирішення проблеми ефективного управління ІТ-інфраструктурою хмарного ЦОД використовувались теорія систем, методи теорії ієрархій, методи математичного програмування, методи дослідження операцій і теорії прийняття рішень, методи математичного та імітаційного моделювання, методи теорії штучного інтелекту, стохастичні і евристичні методи пошуку, методи прогнозування, методи математичної статистики, сервісні моделі хмарних обчислень. Достовірність та обґрунтованість отриманих результатів обумовлені коректним використанням математичного апарату, а також підтверджуються результатами обчислювальних експериментів.

Наукова новизна одержаних результатів. Основний науковий результат дисертаційної роботи полягає у вирішенні наукової проблеми ефективного управління ІТ-інфраструктурою хмарного ЦОД на основі розроблення інформаційної технології, що базується на теоретико-методологічних положеннях, моделях і методах процесів управління ІТ-інфраструктурою як складовою інформаційної системи провайдера.

Основні положення дисертаційної роботи, що визначають її наукову новизну, полягають у наступному:

– *вперше:*

1) розроблено методологію управління ІТ-інфраструктурою хмарного ЦОД на основі операторної форми постановки, аналізу і розв'язання задач управління в умовах невизначеності і змінних навантажень, яка відрізняється вибором стратегій управління в різних режимах роботи хмарного ЦОД за рахунок визначення відповідних моделей через

комбінування критеріїв і обмежень, а також за рахунок визначення відповідних схем реалізації, що дозволяє при управлінні хмарним ЦОД перейти від традиційного підходу з одним визначеним методом до програмно-визначеного підходу з використанням множини відповідних схем реалізації, моделей і методів;

2) розроблено структурно-функціональну модель багаторівневої ієрархічної програмно-визначеної системи управління ІТ-інфраструктурою хмарного ЦОД через поєднання стратегічного і оперативного управління з автоматичним керуванням, а також надання програмно-визначених властивостей, яка дозволяє реалізувати задане управління ресурсами і навантаженням ІТ-інфраструктури з вибором стратегій, плануванням і управлінням їх реалізацією, автоматичним керуванням з урахуванням структури системи, ретроспективних даних її функціонування та дотриманням заданих показників якості угоди про рівень обслуговування;
3) розроблено оригінальну інформаційну технологію управління ІТ-інфраструктурою хмарного ЦОД на основі запропонованої методології, моделей і методів яка відрізняється адаптивністю та використанням стратегічного управління з прогнозуванням за критеріями енергозбереження, забезпечення заданого рівня обслуговування, зниження операційних і капітальних витрат, що дозволяє підвищити ефективність використання ресурсів в умовах змінних навантажень;
4) розроблено адаптивний метод комбінованого прогнозування навантаження на обчислювальні ресурси хмарного ЦОД з використанням усередненого, зваженого та оптимізаційного комбінування оцінок прогнозів, обчислених за альтернативними методами прогнозування та з адаптацією розміру навчальної вибірки, який відрізняється обчисленням та використанням певних типів комбінованих прогнозів (усередненого та зваженого) в реальному часі, отриманих за множиною альтернативних методів, що дає можливість оцінювати високоякісне прогнозоване значення для відомих видів змішаних навантажень в ІТ-інфраструктурі;

5) розроблено метод інтегрованого управління ресурсами ІТ-інфраструктури хмарного ЦОД на основі динамічної моделі його станів із застосуванням стохастичного пошуку для виконання міграцій віртуальних машин, вивільнення, увімкнення і вимкнення фізичних серверів, а також прогнозування для адаптації до змін навантаження з метою досягнення сталого режиму роботи згідно визначених критеріїв, що дозволяє ефективніше управляти ресурсами ІТ-інфраструктури хмарного ЦОД із дотриманням заданих показників якості угоди про рівень обслуговування;
6) розроблено метод управління розподіленим сховищем хмарного ЦОД на основі моделі дворівневого сховища з реплікацією та урахуванням кешування за розміром файлу і за кількістю транзакцій доступу до файлу для управління міграцією і з урахуванням кількості транзакцій доступу до блоків даних, об’єму вільного місця та затримки передачі даних між вузлами зберігання даних для управління реплікацією, який відрізняється застосуванням принципу гіперконвергентності, що забезпечило підвищення рівня надійності збереження даних, відмовостійкості та продуктивності їх оброблення в сучасних апаратно-програмних комплексах ЦОД;
7) розроблено метод управління потужністю хмарного ЦОД на основі динамічної моделі його станів, який використовує запропоновані метрики оцінки стану хмарного ЦОД (коефіцієнт життєздатності віртуальної машини, індикатор дисбалансу фізичного сервера, коефіцієнт відношення необхідних ресурсів до середнього об’єму наявних ресурсів, поріг вільних ресурсів та метрика ємності ЦОД), враховує гетерогенність фізичних серверів, їх тип і кількість, який на відміну від існуючих підвищує якість обслуговування користувачів і зменшує споживання електроенергії, що дає можливість автоматично забезпечити ресурс потрібного типу, а також мінімальну затримку розгортання сервісів у хмарі;

– *отримали подальший розвиток:*

8) декомпозиційно-компенсаційний підхід через надання адаптивності, багаторівневості та програмно-визначених властивостей за рахунок реалізації ефективного управління ресурсами ІТ-інфраструктури хмарного

ЦОД з переходом від вибору стратегій до планування і управління їх втіленням з урахуванням структури системи та її параметрів;

9) алгоритми і методи стохастичного пошуку через розроблення нових методів управління ресурсами ІТ-інфраструктури, а саме методу рівномірної консолідації віртуальних машин з використанням ідеї імітації відпалу, двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку, методу динамічної консолідації і розміщення віртуальних машин на основі алгоритму навчання з підкріпленням, що відрізняються врахуванням процедур визначення станів системи на основі нових метрик, врахуванням достатньої кількості видів ресурсів і дискретності вимірів, що дозволяє ефективно визначати схему реалізації стратегії управління для різних режимів роботи хмарного ЦОД;
10) модель управління ІТ-інфраструктурою з координатором через застосування програмно-визначеного керування підсистемами, де координатор генерує керуючі впливи для програмно-визначених контролерів мережі, сховища і гіпервізорів, погоджуючи їх роботу з визначеною стратегією управління хмарним ЦОД, що дозволило застосовувати її в середовищах з програмно-визначеним функціонуванням.

Практичне значення одержаних результатів. Розв’язані у дисертаційному дослідження завдання, розроблені методи і підходи становлять методичну базу розроблення і реалізації систем управління ІТ-інфраструктурою хмарних ЦОД і підвищення ефективності їх функціонування. Створена інформаційна технологія може бути застосована для розроблення підсистем, компонентів та інших складових систем управління ІТ-інфраструктурою провайдерів хмарних послуг.

До числа результатів, які мають найбільше практичне значення, належать: методологія управління на основі операторної форми вибору стратегії управління і схеми її реалізації; підхід до управління багаторівневою ієрархічною системою ресурсів ІТ-інфраструктури хмарного ЦОД; методи прогнозування навантаження хмарного ЦОД; методи управління ресурсами, навантаженням і потужністю хмарного ЦОД з урахуванням прогнозів; методи

управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД.

Розроблені теоретичні положення, методи і алгоритми використані при: модернізації систем управління функціонуванням інформаційно-телекомунікаційної інфраструктури в компаніях-членах Асоціації «ТЕЛАС»; розробленні системи управління функціонуванням інформаційно-телекомунікаційної інфраструктури ТОВ «АМ ІНТЕГРАТОР ГРУП»; модернізації системи управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО». Впровадження результатів досліджень дозволило на 28% скоротити операційні витрати на управління ІТ-інфраструктурою без порушень SLA; зменшити споживання електроенергії на 17% в середовищі з гомогенними конфігураціями фізичних серверів; скоротити в середньому на 18% кількість фізичних серверів, що обслуговують навантаження клієнтів; зменшити кількість порушень SLA на 27% при наданні ІТ-послуг; скоротити витрати на експлуатацію серверного парку на 19% при забезпеченні виконання заданих вимог угоди про рівень обслуговування.

Теоретичні і практичні результати дисертаційної роботи склали основу нових спецкурсів, що викладаються автором на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України “Київський політехнічний інститут імені Ігоря Сікорського”: “Технології віртуалізації та хмарних обчислень”, “Сучасні технології розроблення програмного забезпечення”, “Інтелектуальні системи управління технічними пристроями”, “Програмування інтернету речей”, “Методи та системи штучного інтелекту”, “Обробка надвеликих масивів інформації”.

Особистий внесок здобувача. Усі наукові результати дисертаційної роботи отримані автором самостійно у друкованих працях: [269, 271, 272] – теоретичне обґрунтування еволюційних процесів розвитку і становлення сучасної ІТ-інфраструктури та підходів до її оптимізації, [280] – двостадійний метод управління консолідацією ВМ хмарного ЦОД, заснований на застосуванні променевого пошуку, [281] – метод консолідації ВМ хмарного ЦОД, заснований на застосуванні модифікованого методу навчання з підкріпленням, [283] –

модель динаміки хмарного ЦОД, яка враховує різні типи ФС, різні типи ВМ і різні типи ресурсів в гетерогенній ІТ-інфраструктурі хмарного ЦОД, [284] – метод інтегрованого управління ресурсами ІТ-інфраструктури ЦОД, що забезпечує необхідну еластичність на рівні ФС з урахуванням споживання енергії та з урахуванням порушень угоди про рівень обслуговування і заснований на моделі ЦОД в просторі станів, моделі енергоспоживання, моделі порушень SLA і методі управління ємністю ЦОД, [286] – адаптивний двоетапний і адаптивний комбінований методи прогнозування споживання обчислювальних ресурсів, які забезпечують меншу помилку прогнозу у порівнянні з методом прогнозу за моделлю, отриманою на основі навчальної вибірки фіксованого розміру, [287, 346] – метод управління і модель дворівневого сховища з реплікацією, [277, 280, 283, 284] – метрики оцінювання стану і динаміки змін ресурсів хмарного ЦОД, [344, 345] – прикладні аспекти використання ІТ-інфраструктури для розміщення і обробки знань в СППР.

У друкованих працях, опублікованих у співавторстві, автору належать: [267, 273] – архітектурні рішення щодо побудови апаратно-програмних комплексів для реалізації прикладних інформаційних процесів, [268, 270] – підхід до оптимізації передачі даних в мережі провайдера хмарних послуг та прикладні аспекти використання цього підходу, [16] – концептуальні основи інформаційної технології управління хмарними ресурсами ІТ-інфраструктури ЦОД, структурно-функціональна модель типової програмно-визначеної системи управління і адаптивний програмно-визначений підхід до розподілу ВМ хмарного ЦОД, який полягає в управлінні фізичними і віртуальними ресурсами через вибір стратегій управління з метою адаптації до зовнішніх впливів, [274] – підхід до управління ІТ-інфраструктурою інтернету речей із забезпеченням заданої якості надання сервісу при раціональному використанні ресурсів ЦОД, методи управління ресурсами екосистеми інтернету речей з архітектурою мікрохмари, [275] – стратегії управління і алгоритми адаптивного програмно-визначеного методу до розподілу ресурсів ІТ-інфраструктури між ВМ хмарного ЦОД, [276] – підхід до організації крайових (Edge) обчислень на основі архітектури системи інтернету речей у вигляді мікрохмари для забезпечення

заданої якості ІТ-послуг з раціональним використанням ІТ-ресурсів, розвиток декомпозиційно-компенсаційного підходу при управлінні ІТ-інфраструктурою екосистеми інтернету речей, [277] – метод динамічного розподілу ресурсів ІТ-інфраструктури між ВМ хмарного ЦОД на основі стохастичного локального пошуку з використанням різних евристик, урахуванням багатовимірності, стохастичності, збалансованого навантаження на ФС, обмеження на кількість одночасних міграцій і обмеження апаратного забезпечення, [278] – метод консолідації ВМ хмарного ЦОД, заснований на застосуванні променевого пошуку, [279] – розвиток принципів координації керуючих впливів при управлінні рівнем послуг в екосистемі інтернету речей із забезпеченням ефективного використання ресурсів ІТ-інфраструктури, [282] – дворівнева система управління гіперконвергентною інфраструктурою ЦОД на основі принципів координації, [285] – двостадійний метод управління ресурсами хмарного ЦОД, заснований на застосуванні променевого пошуку з урахуванням обмежень на кількість одночасних міграцій і з урахуванням енергоспоживання ФС, [343] – прикладні аспекти використання ІТ-інфраструктури для розміщення і обробки наукових публікацій, [348] – підхід до декомпозиції управління технологіями, застосунками і процесами в інформаційній технології, [349] – операторна форма постановки, аналізу і розв’язання задач управління ІТ-інфраструктурою хмарного ЦОД.

Апробація результатів дисертації. Основні результати наукових досліджень доповідалися та обговорювалися на закордонних та міжнародних наукових конференціях і форумах, зокрема: «IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications» (Metz, France, 2019), «Університет і регіон: проблеми сучасної освіти» (м. Луганськ, 2009), «Комп’ютерні науки для інформаційного суспільства» (м. Луганськ, 2010, 2011, 2012, 2013), «Информационно-компьютерные технологии в экономике, образовании и социальной сфере» (м. Сімферополь, 2013), «Комп’ютерні інтелектуальні системи та мережі» (м. Кривий Ріг, 2015, 2016, 2017), «ABIA-2015» (м. Київ, 2015), «Computer Science and Information Technologies (CSIT)» (м. Львів, 2016, 2017, 2018),

«International Conference Radio Electronics & Info Communications (UkrMiCo)» (м. Київ, 2016, м. Одеса, 2017, 2018), «Problems of Infocommunications Science and Technology (PIC S&T)» (м. Харків, 2016, 2017, 2018), «World Forum on Internet of Things (WF-IoT)» (Reston, VA, USA, 2016), «Cloud Computing, GRIDs, and Virtualization» (Athens, Greece, 2017), «Інформатика та обчислювальна техніка (IOT)» (м. Київ, 2017, 2018), «Electrical and Computer Engineering (UKRCON)» (м. Київ, 2017), «Automatic Control and Information Technology (ICACIT'17)» (Cracow, Poland, 2017), «Computer Science, Engineering and Education Applications (ICCSEEA)» (м. Київ, 2018, 2019), «Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)» (Lviv-Slavske, Ukraine, 2018). Результати дисертаційних досліджень доповідались на наукових семінарах: кафедри автоматики та управління в технічних системах, кафедри математичних методів системного аналізу Національного технічного університету України "Київський політехнічний інститут імені Ігоря Сікорського", кафедри інтелектуальних та інформаційних систем Київського національного університету імені Тараса Шевченка.

Публікації. За результатами дисертаційних досліджень опубліковано 71 наукова праця, з них: 1 монографія, 4 статті в закордонних наукових виданнях проіндексовані в системі Scopus, 5 статей опубліковані в закордонних наукових виданнях, 18 статей опубліковані у фахових наукових виданнях України, а також 43 публікації в міжнародних і всеукраїнських конференціях, з яких 12 статей міжнародних конференцій проіндексовані в системі Scopus і цифровій бібліотеці IEEE Xplore.

Структура та обсяг роботи. Дисертаційна робота складається зі вступу, дев'яти розділів, висновків, списку використаних джерел із 353 найменувань та 4 додатків. Загальний обсяг дисертації становить 402 стор., з яких 383 стор. основного тексту, 75 рисунків, 20 таблиць.

РОЗДІЛ 1. АНАЛІЗ СУЧАСНИХ ПРОБЛЕМ УПРАВЛІННЯ ІТ-ІНФРАСТРУКТУРОЮ ХМАРНИХ ЦЕНТРІВ ОБРОБЛЕННЯ ДАНИХ

У розділі виконано аналіз проблеми управління ІТ-інфраструктурою хмарного ЦОД в цілому, а також проблем управління ресурсами і підсистемами. Проаналізовано стан ринку хмарних послуг і послуг ЦОД в Україні і світі, сучасні технології, методи і моделі ієрархічного, інтегрованого та програмно-визначеного управління ресурсами ІТ-інфраструктури ЦОД та інфраструктури інтернету речей на основі методів штучного інтелекту, стохастичного пошуку та прогнозування. На основі системного підходу проаналізовано інфраструктуру хмарних ЦОД як об’єктів керування, сформульовано науково-технічну проблему та задачі дослідження.

1.1. Особливості надання хмарних послуг в Україні і світі

Ринок хмарних послуг в Україні і в світі за останні два роки суттєво зростав [332, 337]. Керівники компаній, що надають хмарні послуги в Україні, висловлюють оптимістичні прогнози щодо його зростання, не зважаючи на те, що в сегменті комерційних ЦОД Українські компанії складають лише 0.05% світового ринку. Подальше зростання ринку хмарних послуг в Україні обумовлено необхідністю розміщення обчислювальних потужностей ближче до споживача і дотримання вимог законодавства. Комерційні ЦОД в Україні заповнені приблизно на 60%, таким чином, є можливість розміщення і реалізації нових хмарних послуг для споживачів в Україні, особливо для малого та середнього бізнесу. При цьому ринок хмарних послуг випереджає ринок комерційних ЦОД.

Наведемо дані про діяльність провайдерів комерційних ЦОД в Україні в 2017 році, рис. 1.1. Щорічний ріст ринку комерційних ЦОД в Україні складає 10%. У 2017 році провайдер «Бі Мобайл» впровадив 400 серверних стійок, «De Novo» - 178 стійок, «Cosmonova» – 35 стійок, «Giga Center» - 28 стійок.

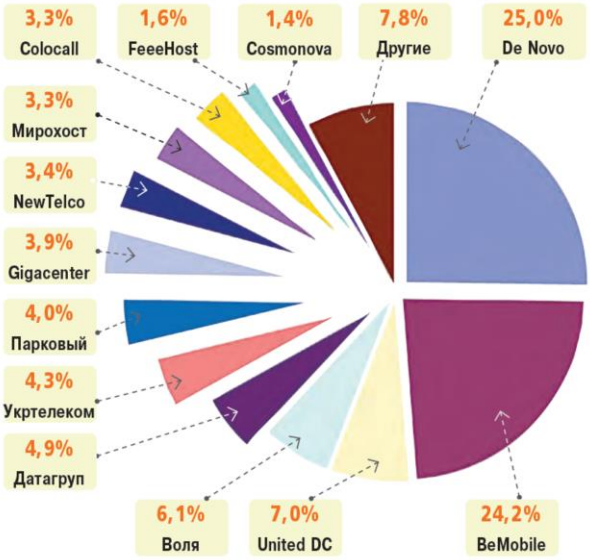


Рис. 1.1. Частки основних гравців українського ринку комерційних ЦОД. Дані «СІБ», липень 2018 року [334]

Динаміка зростання кількості стійок в Українських комерційних ЦОД показана на рис. 1.2. При цьому, провайдери послуг ЦОД використовують для нових потужностей серверні системи на базі ЦП останнього покоління.

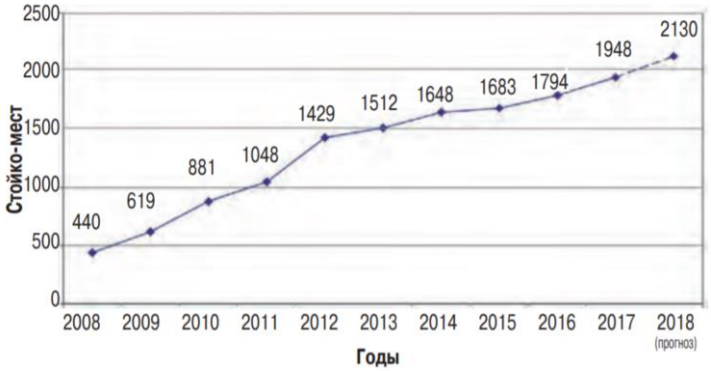


Рис. 1.2. Обсяг українського ринку комерційних ЦОД в умовних стійко-місцях. Дані «СІБ», липень 2018 року

В 2018 році ринок collocation в комерційних ЦОД залишався стабільним, без суттєвих змін. Місцеві ЦОД надають значно дешевші послуги тому частка закордонних провайдерів комерційних ЦОД в Україні складає приблизно 10%. Це обумовлено тим, що в Україні дуже низький рівень повсякденних обчислень для місцевих потреб (бізнес, соціальні сервіси), тобто клієнти нечасто встановлюють серверні системи на площадки комерційних ЦОД, а частіше встановлюють сервери на своїй території [332].

Ринок хмарних послуг в Україні, на відміну від ринку комерційних ЦОД, зростає більш суттєво, і буде зростати на 30-40% щорічно [332]. Не зважаючи на

те, що цифрова трансформація відбувається повільно, споживання хмарних послуг буде зростати і надалі. Основні споживачі хмарних послуг в Україні наведені на рис. 1.3. Сегмент IaaS в Україні складає 0.07% від світового ринку, 65% від всіх хмарних сервісів, що надаються в Україні, і зростає приблизно на 30% кожен рік [333].

Рис. 1.3. Українські споживачі хмарних послуг у 2017 році за сферами діяльності. Дані «СІБ», липень 2018 року [333]

На рис. 1.4 показані основні провайдери хмарних послуг в Україні. За результатами досліджень, наведених в [332], в Україні користується попитом середня конфігурація ВМ серед всіх ресурсів IaaS. При цьому, частка collocation зменшується, а частка хмарних послуг за гібридною моделлю, в тому числі резервне копіювання в хмару, – збільшується.

Хмарні послуги надають, зачасту, ті ж самі компанії, що надають серверні потужності і послугу collocation в комерційних ЦОД. Наприклад, провайдер «UCloud» побудував свою IT-інфраструктуру на базі продуктів VMWare (гіпервізор vSphere ESXi) і Windows Server 20016 (гіпервізор Hyper-V). Він пропонує найбільш потужну ВМ на ринку України – 72 vCPU і 760 ГБ ОП. Провайдер «GigaCloud» дуже широко впроваджує швидкісні накопичувачі даних SSD і будує свою IT-інфраструктуру з використанням продуктів VMWare, OpenStack і Microsoft. Загалом у 2018 році українські провайдери хмарних послуг збільшили показники надання хмарних послуг: провайдер «De Novo» збільшив

надання хмарних послуг на 40%, провайдер «UCloud» - на 70%, провайдер «GigaCloud» - на 400%.

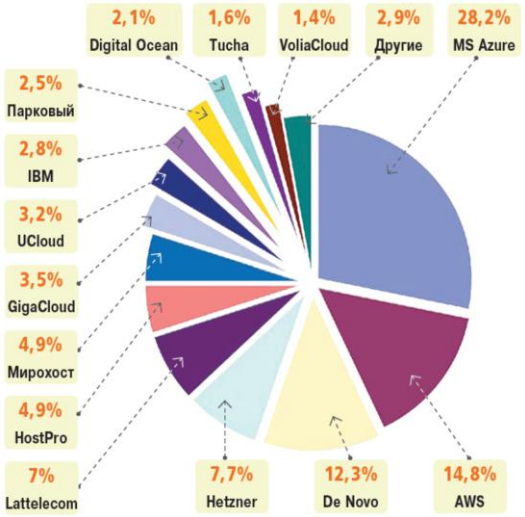


Рис. 1.4. Структура українського ринку хмарних операторів IaaS в 2017 році. Оцінки «СІБ», липень 2018 року [333]

Стосовно глобальних тенденцій розвитку хмарних обчислень і відповідних потужностей ЦОД необхідно звернутись до світових аналітичних компаній та вендорів. Наприклад, згідно звіту Cisco Global Cloud Index (GCI) [335], представленого у 2018 році, з 2016 по 2021 рік обсяг робочих навантажень хмарних обчислень збільшиться в 2,3 рази, при тому, що за той же період навантаження на традиційну інфраструктуру знизиться на 22%. У звіті GCI одним із драйверів ринку наведено сектор Інтернету речей (IoT), оскільки для забезпечення роботи нових рішень IoT необхідні більші обчислювальні потужності, в тому числі віртуалізовані системи класу Edge і хмарні IT-інфраструктури. Згідно цього ж звіту, ЦОД на базі гіперконвергентних систем буде вміщувати до 628 серверів, що на 53% більше, ніж в 2016 році. Крім того, вимоги пропускної здатності в ЦОД подвоюються кожні 12-15 місяців [157]. Згідно дослідження [337], 94% компаній, що взяли участь в опитуванні, використовують хмарні обчислення, при чому 91% з них використовують публічні хмари, а 72% використовують приватні хмари. Також, в середньому, 38% навантаження компанії розміщують в публічній хмарі, а 41% - в приватній хмарі. Більшість навантаження в хмарному середовищі обслуговується ВМ, рис. 1.5.

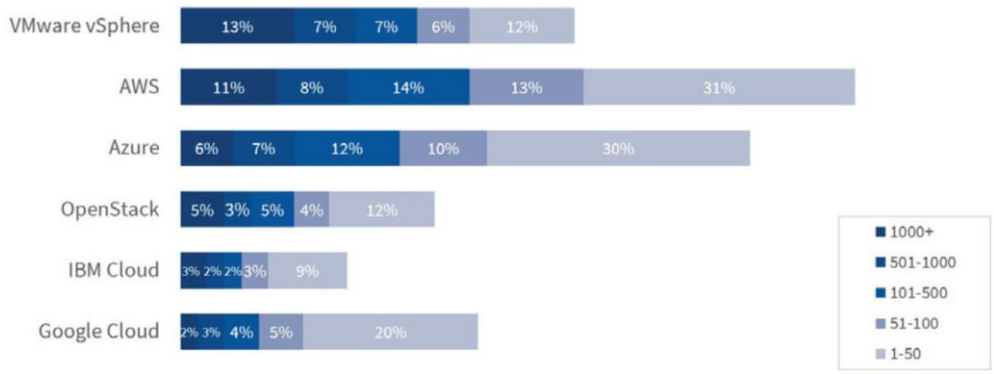


Рис. 1.5. Кількість ВМ, які виконуються в хмарі [337]

На сьогодні найбільше зусиль підприємства малого бізнесу та великі підприємства спрямовують на подолання проблем, які пов'язані з управлінням IT-інфраструктурою в хмарі та зниженням витрат на її утримання, рис. 1.6. З кожним роком проблеми, що виникають при обслуговуванні хмарної IT-інфраструктури, зростають.

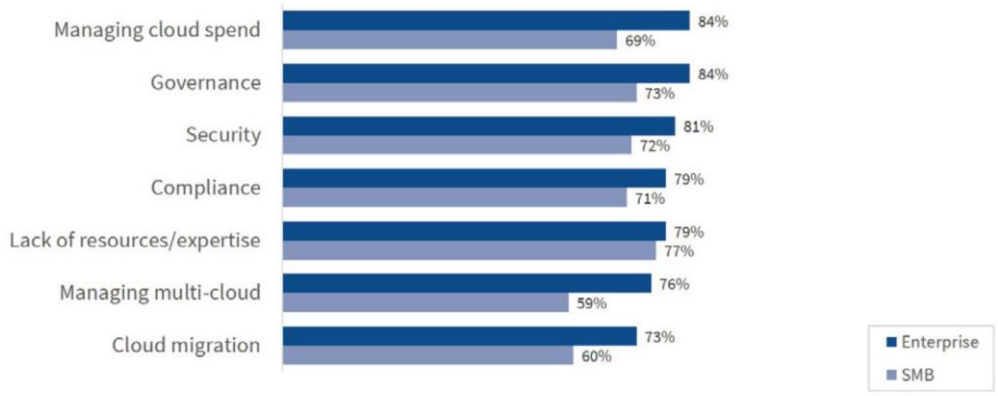


Рис. 1.6. Проблеми, що виникають при обслуговуванні хмарної IT-інфраструктури [337]

З 2017 року клієнти почали впроваджувати більш складні хмарні рішення, в тому числі в гібридній моделі розгортання. Це потребує формулювання і вирішення актуальної науково-практичної проблеми управління IT-інфраструктурою провайдера хмарних послуг з досягненням заданих показників ефективності.

1.2. Хмарний центр оброблення даних як об'єкт керування

Еволюція інформаційних процесів в хмарних ЦОД

Реалізація інформаційних процесів та надання доступу до сучасних інформаційних сервісів пов'язані із використанням ЦОД, або дата-центрів –

комплексних централізованих систем для створення високопродуктивної, відмовостійкої ІТ-інфраструктури. До головних завдань ЦОД належать консолідоване зберігання і опрацювання даних користувачів, підтримка функціонування застосунків та надання користувачам прикладних сервісів на заданому якісному рівні. Основною задачею реалізації інформаційних процесів є надання ІТ-послуг. ЦОД, який серед інших послуг надає послуги згідно сервісних моделей хмарних обчислень, отримав назву хмарний ЦОД.

Еволюційний розвиток інформаційних процесів і архітектур хмарних ЦОД призвів до появи нових понять, доповнив і частково замінив собою поняття класичної теорії інформаційних мереж (ІМ) [338]. Центральне місце в теорії ІМ займає поняття інформаційного процесу (ІП) [339, 340]. Теорія інформаційних процесів, як одного з напрямків кібернетики, почала розвиватися в сервіс орієнтованій архітектурі (SOA) [341]. У сучасних ІМ інформаційні процеси реалізуються так званими ІТ-послугами (IT Service).

Концепція ЦОД втілена багатьма великими компаніями для забезпечення доступу великої кількості користувачів до певних ресурсів (сервісів, застосувань, обчислювальних потужностей, даних тощо). Ефективне управління ЦОД пов’язане з необхідністю розв’язання низки проблем, насамперед створення умов для функціонування інформаційно-обчислювальних потужностей ЦОД, управління віртуалізованими ресурсами, забезпечення надійності та безпеки [269]. Вкладаючи кошти, компанії намагаються збільшити прибуток, покращити якість сервісів, зменшити витрати на експлуатацію ЦОД, знизити вартість обслуговування користувачів, що дозволить, зрештою, закласти основу для ефективної діяльності як самої компанії, так і клієнтів.

Системи управління ІТ-інфраструктурою (СУІ) на операційному рівні забезпечують підтримку функцій, які визначені в еталонній моделі HP ITSM [347]:

1) управління операціями – вимірювання, моніторинг і діагностика подій, що відбуваються при виділенні, використанні і управлінні всіма ресурсами;

2) управління збереженням – управління ємністю, планування, моніторинг і управління завантаженням СЗД з метою раціонального використання ємностей за умови гарантованого забезпечення потреб бізнес-задач;
3) управління споживачами – забезпечення якості їх роботи при доступі до ресурсів IT-інфраструктури, даних і застосунків;
4) служба підтримки – консолідація, реєстрація, пріоритезація, відстеження, управління, ескалація інцидентів і проблем з IT-інфраструктурою;
5) управління рівнем обслуговування і сервісами для бізнесу – визначення і реалізація відповідності SLA між IT-службою і бізнес-підрозділами, описання і управління сервісами з позицій і вимог задач бізнесу;
6) управління ресурсами – облік, оцінювання потреби і розподіл ресурсів (програмних, апаратних та ін.) для надання сервісів на замовлення користувачів;
7) управління змінами і конфігураціями – використання стандартних методів і процедур для здійснення змін в IT-інфраструктурі;
8) управління проблемами і інцидентами;
9) управління потужностями і планування робіт – узгодження потужностей IT-інфраструктури потребам бізнесу і управління її навантаженням;
10) фінансове управління IT – управління фінансовими аспектами IT-операцій;
11) автоматизація IT-процесів – автоматизація потоків робіт IT-служби.

Забезпечення рівня вимог користувачів з мінімізацією витрат становить сутність проблеми управління функціонуванням ЦОД. Зазвичай цю комплексну проблему розбивають на ряд задач менших розмірів. Однією з них є задача управління ресурсами і навантаженням в IT-інфраструктурі хмарного ЦОД [281].

Весь комплекс задач управління IT-інфраструктурою складається з чотирьох стадій: розроблення і впровадження послуг, надання послуг, узгодження бізнесу та IT, управління послугами (рис. 1.7). На оперативному рівні СУІ забезпечують підтримку функцій управління мережами, серверами, БД і застосунками IT-інфраструктури через моніторинг, управління і оптимізацію їх продуктивності.

Рис. 1.7. Комплекс задач управління IT-інфраструктурою [347]

Необхідні гнучкі рішення, які ґрунтуються на оцінюванні стану ресурсів і обсягів навантаження та полягають у правильному розподілі навантаження і ефективному управлінні ресурсами. Для систематичного прийняття правильних рішень необхідні інструментарій та комплекс методик і алгоритмів для розв'язання задач управління IT-інфраструктурою. Їх створення становить важливу науково-практичну проблему, розв'язання якої вимагає розуміння процесів, які відбуваються в хостингових компаніях, функціонування IT-інфраструктури, чіткої постановки конкретних завдань дослідження, розроблення математичних моделей, моделей ЦОД і відповідних методів розв'язання задачі та реалізації згаданих вище методик і алгоритмів [281].

Технології віртуалізації надають можливість абстрагуватися від особливостей окремих груп ресурсів, об'єднати їх у апаратно-програмні комплекси потрібної конфігурації і спростити управління ними [288, 291]. Віртуалізація ресурсів інфраструктури полягає у створенні віртуальних машин (ВМ) – програмних абстракцій, що запускаються на платформі фізичних апаратно-програмних (хостових) систем. Віртуалізація ресурсів узагальнює підходи до створення ВМ і переносить їх на усі види ресурсів, такі як обладнання ЦОД, простори імен, мереж та ін. [205, 213].

Впровадження віртуалізації пов'язано з комплексом додаткових заходів, оскільки процеси прийняття рішень стають складнішими і вимагають розвиненого інструментарію їх підтримки [214, 296, 301]. З'являється

необхідність розроблення моделей, алгоритмів здійснення міграцій ВМ та управління інформаційно-обчислювальними процесами ЦОД з урахуванням специфіки архітектурних рішень, особливостей віртуалізації та їх реалізації у системі управління ІТ-інфраструктурою [292, 298].

Робоче навантаження на хмарні сервіси ЦОД змінюється з часом в широкому діапазоні. Зазвичай, змінюється кількість клієнтів та кількість запитів на виконання послуги або завдання, що потребує одну або декілька ВМ. Для цього потрібно періодично вирішувати задачу оптимізації розміщення нових і перерозподілу існуючих ВМ у хмарному ЦОД. Задача консолідації ВМ [178] є предметом багатьох обмежень, таких як вимоги до ресурсів з боку ВМ, вимоги до безпеки, вимоги щодо доступності та інші.

Управління ресурсами в сервісній моделі хмарних обчислень IaaS

Обчислювальними ресурсами хмарного ЦОД є: процесорний час кожного ФС, оперативна пам'ять кожного ФС, підсистеми збереження даних (локальні і централізовані), мережева підсистема. Управління обчислювальними ресурсами хмарного ЦОД є важливою задачею, розв’язання якої в сучасних умовах забезпечує роботу таких сервісних моделей хмарних обчислень, як інфраструктура як послуга (IaaS), платформа як послуга (PaaS) та застосунки як послуга (SaaS). Для досягнення бажаних показників енергоефективності та продуктивності роботи ІТ інфраструктури в центрах оброблення даних застосовуються технології віртуалізації апаратного забезпечення, програмних засобів, мереж, сховищ даних, робочих місць та інші [280].

Сервісна модель IaaS дозволяє більш ефективно використовувати апаратне забезпечення фізичного сервера (ФС) за рахунок віртуалізації його локальних ресурсів. ФС надає такі ресурси, як процесорний час, пам'ять, локальну підсистему зберігання даних та підсистему доступу до мережі. При цьому, клієнтові надається частина ресурсів ФС у вигляді ВМ або контейнеру. Для реалізації сучасних хмарних послуг клієнт розгортає одну або декілька ВМ необхідної конфігурації, яка визначена провайдером хмарних послуг. Кожній ВМ гіпервізор надає частку ресурсів ФС. З точки зору управління ресурсами хмарного ЦОД, ресурси всіх ФС об’єднуються в *пул* і надаються ВМ для

використання. Таким чином, виникає комплекс задач, пов’язаних з управлінням ресурсами хмарного ЦОД. Один із варіантів перерозподілу ресурсів пулу між ВМ полягає в реалізації процесу консолідації ВМ (англ. virtual machine consolidation) або розміщення ВМ (англ. virtual machine placement). Консолідація ВМ – це розміщення і перерозміщення ВМ на ФС на базі технологій віртуалізації з метою досягнення певних показників ефективності використання ресурсів ЦОД. Фактично, управляючі впливи виробляються для ВМ, ФС, мережевих пристроїв, сховищ, застосунків та інших підсистем [280].

Для роботи кожного екземпляру застосунку в хмарному ЦОД зазвичай створюється окрема ВМ. На відміну від монолітних застосунків, які потребують нарощування ресурсів вертикально (англ. scale-up) при зростанні навантаження, застосунки для хмарних ЦОД (англ. cloud-native applications) потребують нарощування ресурсів горизонтально (англ. scale-out). Горизонтальне нарощування ресурсів полягає у створенні додаткових екземплярів [280] застосунку у вигляді ВМ для обслуговування зростаючого навантаження (запитів клієнтів). Останнім часом, для забезпечення роботи хмарних застосунків широко використовуються так звані контейнери (англ. containers). Контейнер уявляє собою середовище для виконання екземпляра застосунку. Контейнери ізольовані один від одного і виконуються в спільній ВМ або на ФС.

Найбільше поширення отримав підхід до розгортання застосунків на базі ВМ. Виходячи з певних бізнес-потреб клієнт може динамічно змінювати конфігурацію ВМ або налаштовує механізми балансування навантаження, відмовостійкості та резервного копіювання [280]. У процесі роботи хмарного сервісу клієнти створюють множину ВМ. У результаті зміни навантаження, зміни кількості запитів клієнтів та кількості пакетних задач, кількість ВМ повинна змінюватись. Відповідно, змінюється кількість ВМ, що виконуються на окремому ФС, тому деякі ФС виявляються незавантаженими до максимально можливого порогу споживання ресурсів і є недовантаженими, тобто витрачають зайву енергію.

З метою більш ефективного використання обчислювальних ресурсів хмарного ЦОД засоби віртуалізації надають можливість "живої" міграції (англ.

live migration) ВМ з одного ФС на інший. При цьому, вплив на роботу ВМ є мінімальним, і для клієнта міграція ВМ не впливає на виконання задач. Але навантаження на ФС, які обмінюються цією ВМ, зростає [280].

Таким чином, однією з головних задач при управлінні обчислювальними ресурсами хмарного ЦОД є розміщення і перерозміщення ВМ таким чином, щоб задіяти меншу кількість ФС та зменшити кількість міграцій ВМ. Процес перерозподілу ВМ серед ФС, тобто консолідацію ВМ, можливо виконувати як неперервно, так і дискретно. Вирішення задачі консолідації ВМ представлено в багатьох публікаціях [178]. Але дослідження виконувалися для наявних на той час технологій віртуалізації та хмарних технологій. У сучасних ЦОД впроваджуються все нові і нові апаратні та системні засоби, які співіснують з технологіями попередніх поколінь. Таким чином, актуальною є розроблення нових алгоритмів і методів для управління обчислювальними ресурсами ЦОД в цілому, та розміщенням ВМ зокрема.

Для розв’язання задачі управління ресурсами ІТ-інфраструктури хмарного ЦОД пропонується оцінити стан ІТ-інфраструктури за допомогою даних моніторингу, вибрати (або продовжити використовувати вибрану) стратегію управління, отримати завдання і запити користувачів, виконати планування розподілу ресурсів та забезпечення заданої потужності і вирішити відповідні задачі оптимізації перерозподілу ресурсів і навантаження. З цією метою, в дисертаційній роботі розроблені спеціальні метрики для оцінювання стану ЦОД на основі даних моніторингу; розроблений комбінований адаптивний метод прогнозування навантаження на обчислювальні ресурси хмарного ЦОД для планування потужності і розподілу ресурсів; розроблені стратегії управління, які враховують нестачу/надлишок ресурсів та тренд змінення навантаження. Для розв’язання відповідних задач оптимізації перерозподілу ресурсів і навантаження розроблені такі методи: метод інтегрованого управління ресурсами ІТ-інфраструктури хмарного ЦОД; метод управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД; метод рівномірної консолідації ВМ з використанням методу імітації відпалу; двостадійний метод управління ресурсами хмарного ЦОД на основі методу променевого пошуку;

метод динамічної консолідації і розміщення ВМ на основі методу навчання з підкріпленням; метод управління міграцією та розміщенням ВМ в сталому режимі.

Розвиток і поширення гіперконвергентних систем

Еволюціонування і трансформація ЦОД відбувається під впливом таких потужних рушійних факторів і технологій, як віртуалізація [1], розподілені обчислення, GRID [127], хмарні обчислення [60], туманні обчислення (англ. Fog Computing) [128], обчислення великих масштабів (англ. scale-out and warehouse-scale computing) [129, 130], програмно-визначені мережі (SDN) [131, 168, 169] та програмно-визначені ЦОД (SDDC) [132, 133, 165, 166, 167]. Сучасний ЦОД є складною системою, яка розвинулася до рівня хмарних обчислень, що забезпечує значне покращенні самообслуговування, масштабованості та керованості [134].

Дослідження компанії IDC показують, що оператори, які виконують спостереження за функціонуванням традиційної трирівневої IT інфраструктури в ЦОД витрачають більше 70% часу на операції моніторингу, усунення несправностей, пошук вузьких місць при втраті продуктивності, оновлення системного ПЗ і драйверів, переконфігурування підсистем зберігання даних і мережевої взаємодії. Таким чином, підприємства і провайдери послуг при модернізації і розгортанні нових ЦОД поступово переходять від традиційних, багато-вендорних і пропрієтарних IT-інфраструктур до стандартизованих і взаємозамінних рішень, які засновані на програмно-визначеному підході до управління IT-ресурсами [82] і широкому застосуванні віртуалізації, інтегрованих, конвергентних і хмарно-орієнтованих технологій.

На цей час хмарні обчислення стали основною парадигмою, на якій базуються сучасні інформаційні послуги. Підприємства стикаються з об'єктивною необхідністю інтеграції приватних і публічних хмарних рішень. Відповідно досліджень 2017 року компанії RightScale "2017 State of the Cloud Report" [135], 95% організацій запускають застосунки або експериментують з інфраструктурою як послугою, використовуючи гібридну хмару, і головним пріоритетом для 50% IT-компаній є гібридна хмара.

Впровадження приватної хмари підприємства обумовлено в більшості випадків, необхідністю зберігання даних локально (on-premise). Окрім задоволення обмежень на відповідність до регіональних норм, приватна хмара має також значну перевагу над традиційною спеціалізованою інфраструктурою та має низку функцій, які недоступні в публічній хмарі. Однак при збільшенні навантаження на існуючі "наземні" ресурси (on-premise, локальні) виникає необхідність безшовної інтеграції з публічною хмарою для розгортання сервісів, для використання яких бракує локальних ресурсів. Крім того, під впливом вимог до скорочення витрат на електропостачання, капітальних та операційних витрат, продовжує розвиватися і поширюватися інфраструктура віртуальних робочих столів (ВРС, англ. VDI). Також, зростає число користувачів послуг і кількість даних, що виробляються приватними і корпоративними клієнтами хмарних сервісів.

Згідно з дослідженням IDC, одним з основних напрямків модернізації для ЦОД у 2016 і 2017 роках є впровадження модульних IT (Modular IT), які означають модулярізацію IT із застосуванням конвергенції, програмно-визначених систем (SDSys), гіпермасштабування (hyperscale) і контейнерів. За даними IDC [136], до 2020 року великі навантаження на застосунки наступного покоління та нові IT-архітектури у важливих бізнес-об'єктах змусять 55% підприємств модернізувати свої ресурси ЦОД через оновлення існуючих об'єктів та/або розгортання нових об'єктів. Іншим прогнозом полягає в тому, що до кінця 2019 року потреба в поліпшеній гнучкості, кращій керованості та розширеному використанні активів змусить компанії, які ведуть цифрове перетворення, мігрувати більше 50 відсотків своєї IT-інфраструктури ЦОД на використання програмно-визначених моделей обчислень. Крім того, автономна інфраструктура стане стандартною у великих сучасних ЦОД.

Все це сприяє ще більшій консолідації ресурсів ЦОД, зменшенню витрат часу при використанні сервісів управління, адміністрування і конфігурації, і виділенню більшого обсягу часу на створення інноваційних сервісів, що збільшують конкурентоспроможність і прибуток підприємств і провайдерів послуг [147].

Таким чином, виникає необхідність розроблення і дослідження нових архітектур для хмарного ЦОД, які відповідають сучасним викликам, дозволяють адаптуватися до появи нових інформаційних послуг і зміни вимог з боку клієнтів. Одним із напрямків модернізації ЦОД при цьому є дослідження і впровадження гіперконвергентних систем (англ. hyperconverged system). Гперконвергентние системи побудовані з використанням архітектур, підходів і методів, які пройшли стадії впровадження та досліджень на базі ЦОД світових лідерів галузі хмарних обчислень, таких як Amazon, Google, Microsoft, Facebook.

Гіперконвергентна інфраструктура (ГІ) нещодавно отримала значне поширення як середовище, де сервери, сховища та мережі об'єднані за допомогою технологій віртуалізації [161, 163, 164]. Для вирішення сучасних проблем управління хмарними ЦОД [137, 138, 139] гіперконвергентна інфраструктура широко використовується в основній та крайовій IT-інфраструктурі. ГІ – це рішення віртуальної обчислювальної інфраструктури, яке поєднує в собі декілька послуг ЦОД у форм-факторі пристрою та обслуговує різні робочі навантаження, такі як інфраструктура віртуального робочого столу, аналіз великих даних, планування ресурсів підприємства та управління взаємовідносинами з клієнтами.

З огляду на той факт, що багато провайдерів IT-послуг і великі підприємства потребують переходу на більш ефективні, масштабовані і продуктивні рішення, дослідження в області розроблення і використання інтегрованих, конвергентних і гіперконвергентних систем є актуальними. Таким чином, в дисертаційній роботі вирішується задача аналізу архітектурних особливостей та можливостей ГІ з метою їх застосування в умовах приватного і гібридного хмарного рішень, що базуються на сучасних апаратних платформах для IT-інфраструктури ЦОД.

ГІ має принципово іншу архітектуру і відрізняється від традиційних трирівневих інфраструктурних рішень. Гіперконвергенція є архітектурною моделлю IT-інфраструктури, яка використовує віртуалізацію [1], поширені процесори x86, сховища SSD і HDD та мережеву інфраструктуру в одному

будівельному блоці ЦОД. Далі, гіперконвергентні системи зазвичай з'єднуються через мережі на основі протоколів Ethernet і TCP/IP [270, 276, 290].

Сучасна ГІ може масштабуватися до сотень вузлів і поєднувати в собі інженерію веб-масштабу з дизайном клієнтського класу, щоб забезпечити ІТ-інфраструктуру на рівні підприємства. З цією метою в дисертаційній роботі розроблені методики ефективного управління підсистемами ГІ, таких як обчислювальні вузли, сховища даних та мережеві пристрої для надання хмарних сервісів з відповідним масштабуванням.

В дисертаційній роботі досліджено можливість застосування дворівневої схеми координації [249] до управління підсистемами ГІ. Запропоновано використовувати дворівневу координаційну схему для управління підсистемами (контролерами) обчислень, зберігання, мережі та віртуалізації разом з алгоритмами самоврядування в цих підсистемах. Також, проаналізовано архітектурні особливості та можливості ГІ з метою використання їх у приватних та гібридних хмарних рішеннях на основі сучасних апаратних платформ для ІТ-інфраструктури ЦОД.

Аналіз функцій систем збереження даних хмарного ЦОД

Цифрова трансформація, що відбувається в поточний час, впливає на зміни в ІТ інфраструктурі і вимагає впровадження нових технологій і методів зберігання і оброблення даних [31] з метою більш ефективного використання ресурсів та прийняття рішень. Останні тенденції в галузі хмарних обчислень, інтернету речей та машинного навчання спираються на системи збереження даних (сховища даних) як на основний ресурс ЦОД. Управління системами збереження даних (СЗД) становить важливу науково-практичну задачу в умовах віртуалізації та консолідації ресурсів. Для досягнення бажаних показників продуктивності роботи ІТ інфраструктури в цілому необхідно забезпечити надійну, безперебійну та достатньо швидку роботу сховищ даних. Таким чином, виникає необхідність розроблення нових моделей і методів управління системами збереження даних, що покращують продуктивність їх роботи і

дозволяють зменшити капітальні та операційні витрати на IT інфраструктуру в цілому.

Сучасні застосунки виконуються в таких віртуалізованих середовищах, як ВМ і контейнери. Але СЗД підключені безпосередньо до ФС, до мережі ЦОД або до мережі сховищ даних. Тому, з метою управління виділенням необхідних об'ємів сховища із заданими показниками продуктивності і відмовостійкості необхідно віртуалізувати і об'єднати ресурси сховища в пул. Кожна ВМ або контейнер використовується різними бізнес-процесами, які потребують різного об'єму даних в СЗД і різної продуктивності при доступі до даних. Особливістю управління СЗД у віртуалізованому середовищі є необхідність забезпечити різні показники продуктивності роботи зі сховищем даних для різних ВМ та контейнерів на одному ФС.

Важливу роль при роботі застосунків зі сховищем відіграє файлова система. Це обумовлено тим, що такі основні функції, як знімки (snapshots), перевірка цілісності (integrity checking), реплікація (replication), дедублікація (Deduplication), стиснення (Compression) та кодування (erasure coding) реалізовуються на рівні файлової системи, операційної системи або гіпервізора. Файлові системи, що підтримують розподілене зберігання об'єктів, файлів і блоків [50, 51, 45, 52, 53], надають сервісні функції таким чином, щоб забезпечити задану продуктивність і відмовостійкість. Розподілене зберігання даних в більшості файлових систем базується на механізмах реплікації [54]. Одним з основних критеріїв вибору вузлів для розміщення реплік є забезпечення збалансованого (або рівномірного) навантаження на вузли зберігання даних. При цьому рівномірне розподілення реплік можливо досягти як на рівні локальних вузлів, так і на більш високих рівнях, таких як стійка (rack), кластер та ЦОД. Рівномірний розподіл реплік даних, також, повинен забезпечити швидкий доступ до даних з мінімальним часом зчитування (отримання), високу доступність і відмовостійкість. В умовах віртуалізації і хмарних обчислень управління процесом реплікації ускладняється.

Сучасні СЗД являють собою складні підсистеми ЦОД з різними варіантами підключення (DAS, NAS, SAN), побудовані з використанням пристроїв з різними

технологіями запису (флеш-пам’ять, магнітні диски та ін.), і надають доступ до даних, що представлені різними структурами (блоки, файли, об’єкти, документи). Тому, якщо процесорний ресурс має один вимір і може бути розподілений між ВМ або контейнерами, що виконуються локально, то СЗД, як ресурс, має декілька вимірів (об’єм, пропускна здатність, структури даних та ін.). Також, треба врахувати, що навантаження на ВМ та контейнери змінюється, і немає необхідності підтримувати високу пропускну здатність доступу до даних весь час. Це, в свою чергу, призводить до перевантаження одних каналів доступу до даних, а з іншого боку до недовантаження інших каналів.

Ще одним важливим фактором, який впливає на застосування тієї чи іншої СЗД в ЦОД, є ціна. СЗД, що можуть обслуговувати динамічні навантаження з високою продуктивністю і низькою затримкою, побудовані з використанням флеш накопичувачів і коштують на порядки дорожче, чим СЗД, побудовані з використанням накопичувачів з магнітними дисками. Також, на ціну дуже впливає спосіб підключення СЗД до споживачів. Найдорожчим варіантом розгортання СЗД є SAN, набагато дешевшим варіантом реалізації СЗД є DAS.

В решті решт, незалежно від технології організації доступу до сховища, необхідно застосувати той чи інший принцип читання/запису даних. Наявні в ЦОД пристрої, що реалізують старі, нові або найновіші принципи запису даних на носії, можливо розділити за критерієм швидкості на дві групи: швидкісні (найсучасніші) і повільні (пристрої і інтерфейси попередніх поколінь). Зазвичай, впровадження і використання швидкісних пристроїв коштує дорожче (іноді набагато дорожче), чим використання поширених пристроїв зберігання даних попередніх поколінь. Таким чином, завжди виникає задача досягнення високої продуктивності читання/запису даних при зменшенні витрат на придбання і експлуатацію пристроїв і систем збереження даних.

При цьому, ефективне використання ресурсів СЗД полягає у забезпеченні заданої якості обслуговування запитів доступу до даних при мінімізації капітальних та операційних витрат на утримання СЗД.

З появою швидких пристроїв і інтерфейсів в системах збереження даних стало можливим об’єднувати в рамках сховища пристрої з різною

продуктивністю, створюючи так звані багаторівневі сховища (storage tiering) [2, 3, 4, 5 - 32, 33, 34, 35].

Одним із шляхів зменшення вартості придбання і експлуатації СЗД без суттєвої втрати об’єму є міграція даних між швидкодіючими пристроями одного рівня та повільними пристроями іншого рівня сховища. При цьому, рівень швидкодіючих пристроїв має набагато менший об’єм, чим об’єм пристроїв повільного рівня. Відповідно, вартість придбання і експлуатації СЗД зменшується завдяки зменшенню кількості коштовних швидких пристроїв.

Для розв’язання задачі управління СЗД ЦОД в дисертаційній роботі пропонується метод управління, що базується на моделі дворівневого сховища і алгоритмах міграції даних між швидким та повільним рівнями сховища.

1.3. Технології, методи і моделі управління ресурсами ІТ-інфраструктури

На даний час великі хмарні провайдери розробляють і впроваджують технології управління в приватних ЦОД, але потрібні технології управління ресурсами будь-якої ІТ-інфраструктури, що базується на віртуалізації і реалізує принципи хмарних обчислень. Особливо це має значення для таких сфер застосування, як IoT та сховища даних. Ефективне керування ресурсами хмарного ЦОД з урахуванням споживання енергії та дотримання вимог SLA є дуже важливим для провайдера хмарних сервісів. Багато наукових праць за останні роки присвячено методам, підходам та структурам, що призначені для використання в різних умовах функціонування ЦОД з різними програмно-апаратними платформами. Вони мають на меті підвищення ефективності роботи центрів оброблення даних, одночасно мінімізуючи витрати електроенергії та експлуатаційні витрати [107, 108].

Динамічні моделі дискретного часу широко використовуються для моделювання різних процесів при управлінні ІТ-ресурсами [2, 109, 110, 111, 112]. Але в багатьох з них не враховуються або враховуються частково показники, що властиві реальним виробничим системам, такі як гетерогенність ФС, зміна типу і вимог ВМ, облік всіх необхідних ресурсів ФС, обмеження на кількість міграцій та ін.

Дуже важливо зазначити, що при застосуванні методів управління з використанням прогнозу можна досягти високої ефективності при управлінні ресурсами хмарного ЦОД [2, 109, 111, 113]. Дослідження показують, що ресурси ФС у центрах оброблення даних часто використовуються недостатньо через надмірний запас з метою забезпечення обслуговування максимальних (пікових) навантажень або через великі проміжки часу між консолідацією ВМ [106, 73].

Використовуючи інтелектуальні та дискретні моделі, автори [2] розробили загальну структуру забезпечення та алгоритми "зшивання" навантаження, які зменшують енергоспоживання та покращують досвід користувача, беручи до уваги перехідні процеси в ФС та різні алгоритми диспетчеризації навантаження. Але запропонована схема не враховує особливості віртуалізації на базі ВМ.

У роботі [109] автори запропонували використовувати керування з прогнозуванням в динамічній схемі резервування ресурсів для віртуалізованих серверних середовищ. Мінімізація споживання електроенергії досягається через коригування кількості фізичних та віртуальних машин з урахуванням частки процесора та навантаження на кожну ВМ. Метою запропонованої схеми є максимізація кількості транзакцій, які повинні виконуватися з дотриманням SLA, з точки зору середнього часу відгуку при врахуванні витрат на включення та виключення ФС. Однак, вони в основному концентруються на продуктивності серверних застосувань, а не на плануванні розміщення ВМ та ресурсну спроможність ФС.

Автори в роботі [110] пропонують підхід динамічного розподілу ресурсів, який базується на моделях інтелектуального аналізу навантаження. Згідно запропонованого підходу, система керування виділяє або вилучає ФС для роботи застосування на основі оптимізації завантаженості застосування на обмеженому горизонті прогнозування. Головним чином, запропонований підхід орієнтований на хмарну модель обслуговування SaaS і не враховує мінімізацію енергоспоживання. Крім того, метод авторегресії рухомого середнього, який використовується в роботі для прогнозування навантаження, не враховує сезонних змін.

У роботі [114] автори запропонували алгоритм керування на основі ланцюгової моделі Маркова для вирішення проблеми виявлення перевантаження ФС як частини процесу динамічної консолідації ВМ. Для відомого стаціонарного робочого навантаження та заданої конфігурації стану системи політика управління оптимально вирішує задачу виявлення перевантаження ФС через максимізацію середнього часу між міграціями, одночасно дотримуючись QoS вимог. Використовуючи підхід до оцінки навантаження на основі багаторозмірного вікна зі зсувом, автори адаптували модель для оброблення невідомих змінних навантажень. Існує також припущення, що робоче навантаження задовольняє властивості Маркова, що може бути невластивим для всіх типів робочих навантажень. Крім того, вхідні дані для моделювання робочого навантаження, використані в публікації, містять лише навантаження на процесор, якого недостатньо для моделювання гетерогенного середовища.

Питання управління збереженням електроенергії в ЦОД розглянуто в роботі [115], де автори запропонували модель для розв’язання задачі планування в гетерогенних середовищах ФС, щоб досягти мінімального рівня споживання енергії при виконанні всіх типів завдань. Але запропонований підхід базується на використанні верхньої межі інтенсивності надходження всіх типів робочих навантажень, які можуть бути недоступними в реальних ЦОД. Модель споживання електроенергії гетерогенного кластера ФС для обслуговування паралельних завдань запропонована в роботі [116]. Автори представили та оцінили деякі енергоефективні стратегії планування кластеризації щоб скоротити тривалість планування завдання до виконання зі зменшенням споживання електроенергії.

У роботах [111, 112, 113] наведено дослідження щодо методів та підходів динамічного масштабування ємності ресурсів ЦОД. У роботі [113] автори запропонували систему автоматизованого надання ФС, яка передбачає компроміс між енергозбереженням, продуктивністю та надійністю, використовуючи прогнозування та оптимізацію для запропонованої моделі, однак гетерогенність ФС і їх ресурсів у цій роботі не враховані. У роботі [111] запропонована модель для розв’язання задачі динамічного виділення ресурсів.

При цьому мінімізується загальне використання електроенергії при виконанні вимог SLA щодо затримки планування завдання, але запропонована модель розроблена для гомогенного кластеру. Крім того, модель прогнозування потребує налагодження до використання у процесі управління, яке не забезпечується запропонованим підходом. У нещодавній роботі [112] автори запропонували систему управління ресурсами для виконання неоднорідні завдань гетерогенними ФС за критерієм енергозбереження. Запропонована схема враховує гетерогенність і динамічно регулює кількість ФС у відповідності до критеріїв енергозбереження та мінімальної затримки планування виконання, а також з урахуванням витрат на реконфігурацію. Але запропонований підхід використовує ту ж саму модель прогнозування, що і в [111], не розглядає можливість відмов ФС і не враховує штрафних витрат на міграції ВМ.

Аналіз існуючих методів ієрархічного управління

У багатьох попередніх публікаціях декомпозиція хмарного ЦОД виконувалася через поділ рівнів управління кількісним способом, коли верхній рівень керує всіма об'єктами центру оброблення даних, а нижні шари керують групою ФС кластера, стійки або одним сервером [63, 64, 65, 66]. Крім того, в багатьох роботах задача системи управління полягає в управлінні ВМ для забезпечення заданих показників енергоспоживання та забезпечення показників якості надання сервісу на рівні ВМ або сервісу [63, 66, 67].

З точки зору процесу управління, ієрархічні складні системи можуть відрізнятися одна від одної двома основними властивостями: ряд різних процесів працюють протягом різних часових інтервалів, тобто розділені в часі процеси управління [87, 88, 64]; ряд різних процеси поділяють інфраструктуру ЦОД на певним чином визначені групи, що дозволяють ефективно розподілити локальні ресурси, тобто розділені в просторі процеси управління [65, 63, 66].

У роботі [328] автори пропонують керувати не тільки фізичними ресурсами, що надаються для роботи ВМ, але й управляти процесами на рівні операційної платформи та на рівні конфігурацій програмного забезпечення, отримуючи сигнали зворотного зв'язку про показники якості послуг на кожному з цих рівнів.

У роботі [64] автори пропонують ієрархічну систему управління за принципом "знизу вгору" для ефективного управління потужністю, споживаною комп'ютерним кластером. Для ефективного вирішення проблем управління продуктивністю автори розробляють багаторівневу структуру оптимізації, спрямовану на керування кластером на трьох рівнях, де високорівневий контролер L2 керує взаємодією між контролерами нижнього рівня L1 і L0.

Інший підхід до ієрархічного управління ЦОД полягає в тому, щоб розділити домен управління на сфери управління, кожна з яких охоплює підмножину керованих елементів [65]. У [65] автори використовують топологію мережі ЦОД, щоб визначити об’єкти управління, такі як ФС, стійки, кластери та ЦОД в цілому. Мета полягає в тому, щоб використати топологію мережі ЦОД для обмеження потоків даних управління. Але ця робота фокусується на динамічному управлінні ВМ і не враховує показники якості обслуговування на вищих рівнях.

У роботі [66] запропоновано архітектурну основу для керування IT-інфраструктурою великих хмарних ЦОД. Автори пропонують абстрагувати управління ФС в середовищі IaaS на 2 виміри: вимір алгоритмів управління (АУ) і управлінську структуру (УС). Метою запропонованої архітектурної основи є зіставлення виконання робочих навантажень з ФС та іншими вузлами IT-інфраструктури. При цьому АУ відповідає за розв’язання задачі призначення робочих навантажень вузлам інфраструктури і ФС, в той час як УС забезпечує ієрархічне управління та агрегацію метрик, що збираються з цих вузлів.

Автори роботи [68] акцентують увагу на структурі ієрархії управління на основі вимог розміщення програм та сервісів. Запропонована ієрархічна структура автоматично конструює і підтримує ієрархію, яка потім може використовуватися керуючою програмою. Таким чином, запропонована структура використовує поділ на групи при управлінні дочірніми вузлами, визначеними ієрархією. Ієрархія вузлів будується і управляється поверх фізичної мережі ЦОД. У результаті, керуюча програма розміщує програму (або сервіс) користувача на відповідному вузлі або розповсюджує контекст користувача на інші вузли.

Дослідження, що враховують енергетичну складову управління обчислювальними ресурсами, наведені в роботі [67], де автори пропонують рішення на основі ієрархічного управління для реалізації архітектури керування на рівні кластера, що координує декілька контролерів управління енергоспоживанням ФС. Контролер енергоспоживання на рівні ФС визначає рівень його потужності, а контролер більш високого рівня визначає розподіл навантаження між ФС та міграцію ВМ у кластері. Але впливи управління та координації між кількома контролерами енергоспоживання здійснюється без надання гарантій підтримання заданої продуктивності виконуваних програм і сервісів.

У дисертаційній роботі використовуються попередні дослідження авторів [69, 16], а також розроблена багатошарова ієрархічна система управління для управління інформаційними послугами хмарного ЦОД за критерієм мінімізації витрат і мінімізації відхилень параметрів якості надання послуг від заданих. З цією метою пропонується використовувати підхід декомпозиції до розроблення моделі хмарного ЦОД на основі трьох шарів: шар інфраструктури, шар платформи та шар застосунків. Таким чином, багатошарова ієрархічна система управління дозволяє оцінити ефективність управління послугами за якістю наданих послуг, за витратами на використання ресурсів ЦОД і за вартістю порушень SLA на кожному шарі ієрархії.

Аналіз сучасних підходів до управління ресурсами ЦОД з використанням програмно-визначених технологій

У роботах [215, 108, 216] зазначається, що на цей час вченими і технологічними компаніями додаються значні зусилля для розроблення методів, алгоритмів і систем управління обчислювальними ресурсами ЦОД, що надають хмарні послуги. Причому дослідження зосереджені в основному в напрямах надання, розподілу, планування ємності, зіставлення і масштабування ресурсів хмарних і віртуалізованих ЦОД.

При цьому запропоновано велику кількість систем з використанням програмно-визначених технологій і програмних платформ для надання та моніторингу ресурсів, включаючи точні методи, евристики, механізми

передбачення навантаження, методи обліку розбалансованості навантаження ФС, взаємовпливу ВМ, розташованих на одному ФС та ін. Більшість із запропонованих підходів для управління ресурсами в хмарних і віртуалізованих ЦОД засновані на використанні принципу консолідації ресурсів і ВМ, а також на використанні мінімальної кількості ФС при обслуговуванні запитів клієнтів. Причому ФС, на яких не залишилося працюючих ВМ, переводяться в режим мінімального енергоспоживання або відключаються.

Автори роботи [242] запропонували алгоритм керування, який базується на вимірі, прогнозуванні і перепризначенні навантаження на ФС через міграцію ВМ, щоб мінімізувати кількість ФС, необхідних для підтримки робочого навантаження за встановленою нормою підтримки заданого рівня обслуговування SLA. У цьому підході використовується оціночна модель для прогнозування майбутнього попиту на ресурси, а алгоритм керування – для призначення набору ВМ на визначені ФС. Але цей підхід не враховує кількість поточних міграцій.

У роботі [81] запропоновано концепцію для створення технологій глобального обміну інформацією в хмарі і архітектури, орієнтованої на ринок розподілу ресурсів в хмарах, заснованих на необхідності зближення конкуруючих IT-парадигм. Щоб уможливити успішне впровадження хмарних обчислень, автори представили різні варіанти вирішення проблеми, щоб на практиці виявити існуючий потенціал, наприклад, мета-переговорів інфраструктури для глобальних хмарних обмінів та високоефективної доставки контенту через "зберігання в хмарі".

Для управління ресурсами і ВМ в режимі онлайн з урахуванням динаміки змін середовища в сучасних хмарних рішеннях широко застосовується механізм міграції ВМ [240]. У роботі [241] запропонована система для автоматизації моніторингу, виявлення перевантажених елементів IT-інфраструктури, визначення нового плану розміщення ВМ на ФС і виконання міграції ВМ згідно з цим планом. Однак застосовувані в цій системі фіксовані пороги для визначення ступеня утилізації ресурсів не дозволяють ефективно управляти

ресурсами в умовах змішаних навантажень і нестаціонарних моделей використання ресурсів.

У роботі [240] запропонований алгоритм управління міграцією ВМ, що мінімізує число ФС, необхідних для обслуговування навантажень на зазначеному в SLA рівні. Автори використовують оцінну модель для звичайного передбачення необхідності майбутнього споживання ресурсів, тому їх алгоритм не враховує число міграцій ВМ. Автоматизована система управління ємністю і навантаженням, яка інтегрує кілька контролерів ресурсів в трьох різних областях і часових масштабах запропонована в роботі [243].

Автори роботи [244] запропонували систему, що дозволяє адаптивно підлаштовувати виділені для різних ВМ ресурси в умовах хмарних обчислень. Система використовує онлайн модель передбачення потреби в ресурсах для визначення короткострокових потреб у ресурсах. Запропонований підхід використовує спеціальну програму всередині кожної ВМ, що неприпустимо для багатьох систем і платформ хмарних обчислень.

Докладний аналіз проблеми використання енергії та ефективної динамічної консолідації ВМ зроблено в роботі [194]. Автори аналізують різні алгоритми (онлайн, офлайн, детерміновані і динамічні) для розв'язання задачі консолідації ВМ і запропонували адаптивні евристики для процесу динамічної консолідації ВМ, що складається з чотирьох частин: (1) визначення, коли ФС (англ. host) розглядається як перевантажений і вимагає перенесення однієї або декількох ВМ з цього ФС; (2) визначення, коли ФС розглядається як такий, що недовантажений і необхідно перенести всі ВМ з цього ФС, і перемкнути ФС в режим сну; (3) вибір ВМ, які повинні бути перенесені з перевантаженого ФС; і (4) знайти нове місце для ВМ, обраних для міграції з перевантажених і недовантажених ФС. Автори показують, що використання адаптивних евристик для вирішення завдання динамічної консолідації ВМ з високою енергоефективністю і продуктивністю випереджають оптимальний онлайн детермінований алгоритм. Запропонована модель системи управління є багаторівневою і включає локальні і глобальні менеджери. На виході запропонованого алгоритму є комбінована карта міграцій, яка містить інформацію про нове розміщення ВМ, що були перенесені з

перевантажених і недовантажених серверів. Таким чином, процес міграції виконується в дискретному режимі через задані інтервали часу.

Автори роботи [245] запропонували ефективний підхід до консолідації ВМ, який враховує вимоги QoS і може скоротити споживання енергії одночасно із запобіганням порушень SLA. Вони ввели ефективний алгоритм розподілу ресурсів з врахуванням QoS, який враховує два параметри: завантаження процесора ФС і кореляцію використання ресурсів ВМ з іншими ВМ на ФС.

У роботі [246] запропоновано фреймворк відкритого інтерфейсу хмарних обчислень, який підтримує виділення ресурсів, моніторинг і автоматичну конфігурацію для хмарних ресурсів для задоволення вимог застосувань на рівні інфраструктури. Фреймворк включає в себе набір протоколів і API платформи, незалежної від постачальника, яка може вирішувати різні завдання управління при задоволенні вимог до інтеграції, переносимості та сумісності. Нова архітектура розвинула реалізацію платформи агентів JADE від Cloud Agency в якості середовища виконання, яке підтримує зв'язок агентів через канал зв'язку між агентами та стандартну мову комунікацій між агентами на основі HTTP.

У дослідженні [247] представлена мультиагентна система DRPM (Dynamic Resources Provisioning and Monitoring) яка призначена для управління хмарними ресурсами провайдера з дотриманням вимог SLA. Система включає в себе алгоритм для визначення ВМ, яка є претендентом на міграцію. При цьому використовуються глобальний агент і множина локальних агентів. Модель системи DRPM розроблена в середовищі моделювання CloudSim, яка широко використовується для оцінки роботи алгоритмів і методів управління ресурсами в умовах віртуалізації і хмарних обчислень. Результати моделювання показали, що система DRPM дозволяє збільшити утилізацію ресурсів провайдера, знизити споживання електроенергії і уникнути порушень вимог SLA. Однак система працює в дискретному режимі, складаючи план міграції через фіксовані проміжки часу через реалізацію трьох фаз: моніторинг, аналіз, виконання. Крім того, використання локальних агентів всередині ВМ часто неприпустимо в умовах хмарних обчислень.

Програмно-визначена парадигма дає можливість контролювати широкий спектр хмарних ресурсів в динамічному режимі через відокремлення функцій управління від обладнання IT-інфраструктури. Нині повна реалізація функцій програмно-визначеної системи є недосконалою. Проте, є деякі успішні рішення, що наближаються до реалізації повністю інтегрованої програмно- визначеної системи.

Одна з перших спроб вирішення практичної реалізації програмно-визначеної системи для хмарних обчислень наведена в роботі [83]. Автори в [83] представили архітектуру Software Defined Cloud (SDCloud), що представляє економічно ефективну реалізацію визначених користувачем віртуальних інфраструктур в хмарному середовищі. Архітектура SDCloud складається з чотирьох окремих шарів: шар користувача, шар прикладного рівня, шар управління, і шар інфраструктури. Автори оцінювали орієнтований на QoS розподіл смуги пропускання і орієнтовані на мережу сценарії розміщення ВМ за допомогою інструментарію CloudSim і його розширення для підтримки моделювання SDCloud. Автори розглядають різні застосунки і сервіси, що працюють в хмарі, зосередивши увагу на застосуваннях, що реалізують інтенсивні обчислення і обробку даних і використовуються в Web, мобільних і корпоративних середовищах. Проте, автори зосередилися в основному на SDN, програмованих проміжних сервісах, віртуалізації мережевих функцій і передбачає реалізацію програмно-визначених функцій управління в шарі користувальницького управління. Запропонована архітектура SDCloud є спробою подальшого дослідження можливостей програмно-визначених систем. У дисертаційній роботі пропонується інтеграція програмно-визначених мереж, обчислень і систем зберігання на основі існуючих протоколів та інтерфейсів API поширених програмно-визначених контролерів.

Автори роботи [248] запропонували новий експериментальний фреймворк для програмно-визначених ЦОД, що забезпечує віртуалізоване середовище для реалізації тестового стенда програмно-визначеної системи. Пропонована структура заснована на симуляторі Mininet. У той же час основні Mininet компоненти, такі як хост (ФС), комутатор і контролер, були налаштовані для

створення експериментальної бази для моделювання програмно-визначених ЦОД. Пропонована система дозволяє користувачам і дослідникам розробляти і оцінювати різні програмно-визначені рішення.

Необхідно відзначити, що технологічні компанії, що розробляють рішення для хмарних ЦОД, широко застосовують політики управління. У широкому розумінні, політики реалізують частину (декілька функцій) управління процесами в IT-інфраструктурі і є спрощеними засобами вироблення керуючих впливів, комбінуючи які адміністратори досягають певної ефективності роботи системи управління. Тому, для вирішення інтегрованих задач на декількох рівнях управління в ієрархічних системах використання політик ускладнює розв'язання задачі управління і може заважати розв'язанню оптимізаційних задач в окремих підсистемах. Таким чином, політики можуть бути застосовані в сталому режимі роботи ЦОД, коли робота ІП і сервісів не впливає на зміни в структурі і налаштуваннях декількох підсистем (сховищах, мережевому обладнанні, програмно-визначених контролерах).

У попередніх дослідженнях авторів [69, 249, 250, 251, 212, 261] розроблені теоретичні та практичні основи створення систем управління IT-інфраструктурою, включаючи декомпозиційно-компенсаційний підхід, дворівневу модель системи управління ресурсами з координатором, адаптивний генетичний алгоритм і концепцію управління корпоративною IT-інфраструктурою. У дисертації продовжені теоретичні дослідження в галузі управління ресурсами ЦОД з акцентом на розроблення алгоритмів і методів управління ресурсами і ВМ, що володіють властивістю адаптації до збурюючих впливів, з урахуванням динаміки розгортання нових ВМ і процесів міграції ВМ, які відбуваються одночасно, а також з урахуванням тенденцій потреби в ресурсах для досягнення енергетичної ефективності.

У дисертаційній роботі продовжено теоретичні дослідження в напрямку управління ресурсами хмарних ЦОД через розроблення ефективних методів та алгоритмів, включаючи: використання відповідних методів і алгоритмів при розв'язанні оптимізаційних задач з адаптацією до збурюючих впливів; врахування динаміки процесів розгортання нових ВМ та процесів міграції

існуючих ВМ, що відбуваються одночасно; врахування прогнозу попиту обчислювальних ресурсів для досягнення енергоефективності і зменшення порушень умов SLA; використання існуючих політик управління обчислювальними ресурсами та ВМ в сталому режимі роботи хмарного ЦОД.

Аналіз стану систем збереження даних в хмарних ЦОД

Системам збереження даних в хмарних ЦОД приділяється все більше уваги у зв’язку із зміною вимог до швидкісних і об’ємних показників при роботі з даними з боку сучасних сервісів. Останнім часом набули розвитку нові технології роботи з даними, такі як сховища на основі флеш пам’яті (Flash storage), енергонезалежна пам'ять (Non-volatile memory express) [36], фабрики сховищ на основі Ethernet (Ethernet storage fabric) [37], пам'ять класу сховища (Storage class memory) [8-38]. При цьому, адаптація і впровадження нових, більш продуктивних технологій збереження даних не витісняє існуючі, більш повільні, а доповнює їх. Крім того, з одного боку, сучасні сервіси вимагають швидкого доступу до даних і високої продуктивності, з іншого – великих об’ємів для довготривалого зберігання, забезпечення відмовостійкості і резервного копіювання, рис. 1.8.

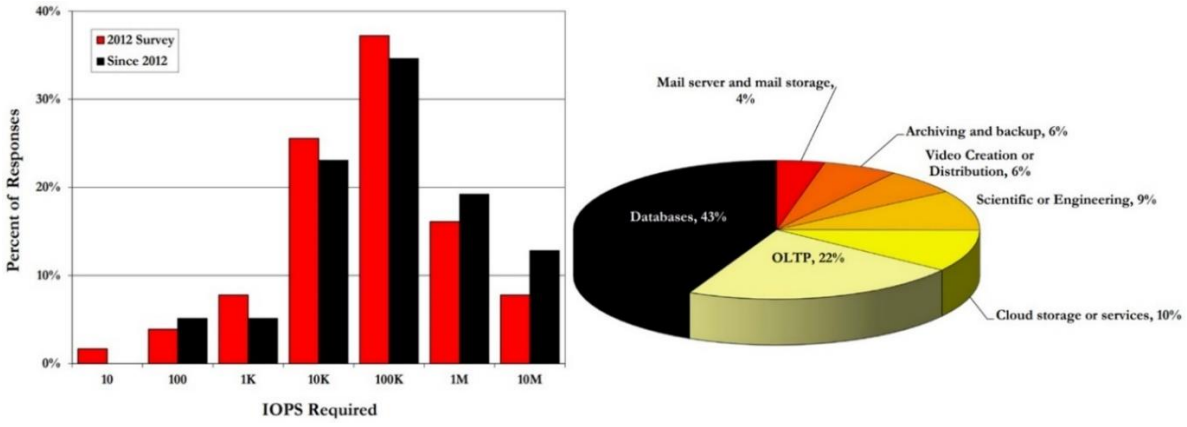


Рис. 1.8. Необхідна продуктивність та використання СЗД застосунками [336]

Розробленню нових методів і фреймворків для підвищення ефективності роботи СЗД приділяється багато уваги з боку науковців та інженерів [33, 34, 35, 39, 40, 41, 42]. У статті [39] запропонований менеджер розміщення даних “AutoTiering”, який працює з багаторівневим сховищем, що складається тільки з SSD пристроїв. Розподіл пристроїв між рівнями сховища відбувається залежно від продуктивності накопичувачів та їх вартості. Стан кожного рівня сховища

оцінюється з використанням метрик: продуктивність операцій читання/запису випадкових даних (IOPS), швидкість читання/запису послідовних даних (MBPS) та розмір сховища. Навантаження для оцінки роботи запропонованого рішення згенеровані з використанням IOMeter [43] і FIO [44]. Основна мета, яку автори [39] при роботі з багаторівневим сховищем намагаються досягти – збільшення прибутків при збережені даних. Максимізація критерію досягається через міграцію даних між пристроями рівнів залежно від вартості збереження даних та вартості їх міжрівневої міграції. Однак, додавання чергового рівня до сховища призводить до необхідності додавати ще один рівень управління і ще один рівень міграції даних, ускладнюючи загальну схему роботи з даними.

У роботі [40] представлено математичні моделі для задач кластеризації та управління ресурсами систем збереження даних з метою мінімізації витрат на користування існуючими сховищами, покращення якості обслуговування споживачів даних та рівномірного розподілення файлів між рівнями збереження даних. Для розроблення моделей систем збереження даних запропоновано використовувати методи математичного програмування з використанням критеріїв мінімізації витрат і рівномірного заповнення рівнів та пристроїв з урахуванням ресурсних, технологічних і часових обмежень. Але розроблені алгоритми показали різну продуктивність роботи з файлами різного розміру. Крім того, не обґрунтовано використання трьох рівнів сховища і не вказано, для якого навантаження проводились дослідження роботи запропонованих алгоритмів.

Схема збереження даних з використанням дворівневого сховища запропонована в роботі [41]. На рівні файлової системи створюються логічні гібридні диски з відповідними позначками розміщення блоків даних, що належать програмам при їх запуску і функціонуванні. Це дозволяє усунути затримки при пошуку блоків даних у відповідних таблицях відображення і розмістити часто використовувані блоки на пристроях з високою продуктивністю. Унаслідок застосування представленої схеми прискорився запуск програм та підвищилась продуктивність роботи зі сховищем. Запропонована схема підтримує міграцію блоків між твердотілими

накопичувачами (solid-state drive, SSD) та пристроями на магнітних дисках (hard disk drive, HDD), та навпаки. Представлений графічний розподіл запитів блоків різної довжини при використані програм LibreOffice та Eclipse показав, що більш частіше відбувається зчитування коротших послідовностей блоків даних. Однак в запропонованій схемі виконується міграція тільки певних блоків, а не всіх, що належать до виконуваної програми. Також, зворотна міграція блоків з рівня SSD на рівень HDD виконується тільки для блоків, які довго не використовуються, незалежно від їх розміру.

Розподілена файлова система, що враховує показники продуктивності різних рівнів системи збереження даних, представлена в роботі [42]. Запропонована файлова система розроблена на базі HDFS [45] і дозволяє застосовувати політики автоматизації управління даними як в межах рівнів сховища на одному вузлі, так і в розподіленому варіанті. Із застосуванням багатокритеріальної оптимізації автори запропонували схеми забезпечення відмовостійкості, балансування навантаження та збільшення продуктивності. Однак запропонована система працює на файловому рівні і орієнтована на вузьке коло застосування, наприклад Hadoop [46] і Spark [47].

Механізм реплікації досліджується, вдосконалюється і адаптується до різних умов застосування у праці [55]. Основними напрямками покращення реплікації є забезпечення надійності зберігання даних, рівномірного розподілу даних по вузлах реплікації, рівномірного завантаження вузлів зберігання даних та високої швидкості отримання даних.

У роботі [56] запропонований алгоритм пошуку вузлів зберігання даних для розміщення реплік в файловій системі HDFS. З метою рівномірного розміщення блоків даних метод інтелектуального розміщення даних обирає вузли кластера, що найменше завантажені. Спочатку метод обирає найменше завантажену стійку, а потім найменше завантажений вузол в цій стойці. Однак запропонований алгоритм не враховує додавання нових вузлів і спирається тільки на оцінювання вільного простору вузла. Крім того, не враховується затримка передачі даних між вузлами. Не уточняється, також, яке саме навантаження на вузол враховується при роботі алгоритму.

Покращена стратегія динамічного пошуку вузла зберігання даних для розміщення репліки запропонована в [57]. Стратегія враховує "температуру" файла і завантаження вузла, а також орієнтована гібридне хмарне середовище і ставить за мету рівномірне завантаження вузлів. Середнє значення "температури" файлу та завантаження вузла використовується для визначення кількості реплік файлу. При цьому вважається, що файл, розташований на вузлі реплікації повністю. Основна увага приділяється визначенню кількості копій файлу. Але запропонована стратегія орієнтована на файли, а не блоки. Крім того, не враховується вільного простору на вузлі зберігання даних і затримка передачі даних між вузлом, зберігає файл і вузлами-репліками.

У роботі [58] запропонована адаптивна стратегія реплікації файлів в ЦОД, яка при виборі вузла реплікації враховує доступність файлу, час останнього доступу до файлу, кількість доступів до файлу та його розмір. Основним критерієм при виборі вузла зберігання даних є досягнення збалансованого завантаження всіх вузлів ЦОД. Але, запропонована стратегія орієнтована тільки на файли, і в умовах, коли розмір файлів дуже відрізняється, забезпечити рівномірний розподіл реплік за критеріями розміру та частоти доступу неможливо. Крім того, в роботі алгоритму не враховано забезпечення цілісності даних і час запису файлів на вузли репліки.

1.4. Практика застосування математичних моделей і методів для розв’язання задач управління хмарним центром оброблення даних

Сучасний хмарний ЦОД є складною системою, що використовує технології віртуалізації, хмарні сервісні моделі IaaS, PaaS, SaaS, програмно-визначені технології і велику кількість різнорідних ресурсів. У середовищі хмарного ЦОД постачальник послуг стикається з різними проблемами, спричиненими розширенням переліку вимог клієнтів та появою нових хмарних послуг. З одного боку, постачальники хмарних сервісів повинні забезпечувати параметри обслуговування QoS (наприклад, час відгуку, затримку, пропускну здатність, простір для збереження даних), покращуючи використання ресурсів і енергоефективність при виконанні SLA. З іншого боку, є ресурсне забезпечення,

розподіл ресурсів, відображення ресурсів на навантаження, масштабування ресурсів, оцінка ресурсів і завдання посередництва ресурсів для забезпечення високоякісної роботи прикладних служб з мінімальною кількістю задіяних ресурсів.

Проблема розподілу ресурсів є важливою і актуальною проблемою сучасних хмарних ЦОД. Сервісна модель IaaS дозволяє клієнтам динамічно запитувати необхідну кількість ВМ на основі їхніх бізнес-вимог. Одним з основних завдань управління ресурсами в IaaS є створення ВМ на ФС хмарного ЦОД. Процес розподілу ВМ між ФС повинен бути виконаний з мінімальною кількістю задіяних ФС та зменшенням споживання енергії операційними задачами. Процес вибору, яку ВМ слід розміщувати на кожному ФС, визначений вище як задача консолідації ВМ. Проблема розміщення ВМ широко вивчена в наукових публікаціях [178]. Ця проблема є предметом багатьох обмежень, що походять з декількох аспектів, таких як вимоги до ресурсів ВМ, вимоги до безпеки, вимоги щодо доступності та інші. Багато обмежень, також, можуть бути визначені в угоді SLA, пов'язаної з ВМ кожного клієнта [179].

Задача консолідації ВМ розглядається в дисертації як багатовимірна задача про пакування в ємності (multi-dimensional bin-packing problem), беручи до уваги, що властивості елементів можуть бути змінені, нові елементи можуть бути замовлені до розгортання, і існуючі елементи можуть потребувати перепризначення на інші ємності. Робоче навантаження у хмарному ЦОД може змінюватися з часом. Клієнти, які запитують послуги або виконання завдань, використовують одну або декілька ВМ. Для цього необхідно періодично вирішувати завдання оптимізації для розгортання нових і перерозподілу існуючих ВМ між ФС. Перепризначення нових і існуючих ВМ повинно виконуватися періодично в асинхронному режимі за потреби. Для розв’язання такої складної комбінаторної задачі оптимізації в дисертаційній роботі пропонується метод стохастичного локального пошуку.

Для розв’язання задачі динамічної консолідації ВМ з рівномірним розподілом навантаження на процесор, оперативну пам'ять та мережевий інтерфейс з метою досягнення максимальної продуктивності та повного

використання ресурсів хмарного ЦОД в дисертації модифікований і застосований алгоритм імітаційного відпалу. Як показано у працях [180, 181, 182], стохастичні методи локального пошуку можуть бути застосовані до проблем керування хмарними ресурсами, при цьому, ці методи знаходять прийнятні рішення в допустимому обсязі часу і ресурсів.

Початкове розміщення ВМ обчислюється з використанням евристик, таких як First Fit Decreasing (FFD) або Best Fit Decreasing (BFD). У дисертаційній роботі використовується евристика BFD з сортуванням усіх ВМ в порядку зменшення їх поточного використання ЦП. У той же час, початкове розміщення ВМ не враховує використання оперативної пам'яті, швидкості та ємності сховища, завантаження мережевого інтерфейсу кожного ФС. Для отримання оптимальної карти розподілу ВМ між ФС з більш збалансованим навантаженням процесора, оперативної пам'яті та мережевого інтерфейсу кожного ФС у процесі оптимізації модифікований і використаний алгоритм імітації відпалу.

Для проведення повторюваних експериментів з дослідженням різних алгоритмів управління ресурсами ЦОД широко використовуються різні симуляційні середовища [183]. Багато результатів досліджень отримані за допомогою методів моделювання, оскільки проводити експерименти на хмарній інфраструктурі ЦОД дуже дорого. Таким чином, використання інструментів моделювання та симуляційних середовищ є корисним і поширеним підходом у при виконанні досліджень в сфері хмарних обчислень. У статті [184] пропонується інструментарій CloudSim для підтримки проектування та моделювання компонентів хмари, таких як ЦОД, ВМ, ФС та алгоритми управління, які можна налаштувати або створити відповідно до вимог досліджень.

Таким чином, актуальною є проблема розроблення і дослідження алгоритмів на базі імітації відпалу для вирішення задачі динамічної консолідації ВМ як складової проблеми управління обчислювальними ресурсами IT-інфраструктури хмарного ЦОД.

Протягом останніх років запропоновано різні підходи для розв’язання задачі розміщення ВМ у хмарному ЦОД [178, 185, 186, 266], яка являє собою

задачу оптимізації з різними цільовими функціями. У багатьох роботах висвітлено цю задачу і запропоновано різні рішення для пошуку оптимального плану розміщення ВМ на ФС з метою мінімізації кількості використаних ФС та забезпечення виконання умов SLA. Більшість з цих праць використовують алгоритми впорядкування та методи прогнозування для створення карти розподілу ВМ на ФС.

Для розв'язання задачі розміщення ВМ широко використовуються мета-евристики, такі як: Табу-пошук [187], еволюційні алгоритми [181], мурашиний алгоритм [180], пошук сусідів [188], імітаційний відпал [182] та ін. Причиною використання локальних алгоритмів пошуку є поліпшення результатів, отриманих такими евристиками як FFD, BFD та їх модифікаціями. Існує декілька цільових функцій, які використовуються при розв’язанні задачі розміщення ВМ, включаючи мінімізацію споживання енергії [189], мінімізацію кількості увімкнутих ФС [190], мінімізацію мережевого трафіку [191], Максимізацію доступності [192], максимізацію використання ресурсів [189] та інші.

Протягом останнього десятиріччя запропоновано багато підходів до розв’язання задачі розміщення ВМ на ФС. У [193] автори пропонують ефективну метаевристику ітераційного локального пошуку для вирішення проблеми багатокритеріальної упаковки ємності і вирішення проблеми перепризначення ВМ на ФС. Запропонований підхід покладається на такі параметри, як перевищення порогових значень контрольованих параметрів, перезавантаження шейкера (оператора струсу), розмір оператора струсу і критерії обрізання. Запропонований підхід виробляє якісні рішення в допустимий час обчислень для масштабних випадків вирішення обох задач. У більшості випадків більше ніж 99% поліпшень порівняно з вихідною конфігурацією розміщення отримані запропонованим способом. Однак, існує два основні недоліки запропонованого підходу. По-перше, не враховано, що міграції ВМ виконуються одночасно без будь-якої затримки в часі, а по-друге, запропонований підхід не враховує обмеження на максимальну кількість одночасних міграцій з/на ФС.

Для вирішення проблеми консолідації ВМ автори роботи [187] запропонували двоетапний алгоритм, який мінімізує кількість необхідних ФС, одночасно гарантуючи, що міграції, виконані при переході на нову схему призначень ВМ на ФС, будуть завершені не пізніше заданого часу. Перший етап спрямований на пошук можливої схеми розміщення ВМ на ФС, що мінімізує час максимальної міграції для всіх ВМ. Друга фаза використовує метаевристичний пошук табу для отримання рішень, що використовують меншу кількість ФС, але також з дотриманням граничного часу міграції. Однак автори не визначили, як виявити перевантажені ФС і не розглянули інші ресурси ФС.

Механізм багатокритеріальної оптимізації для консолідації ВМ розроблений і реалізований в [188]. Автори також запропонували механізм, що дозволяє адміністраторам ЦОД додавати інші цілі оптимізації. Запропонований механізм дозволяє зменшити обчислювальні витрати, класифікуючи ФС у відносно невелику кількість еквівалентних наборів на основі статусу кожного з них. Він також реалізує пошук сусідів, щоб знайти найменш витратне розміщення ВМ при обслуговуванні запиту клієнта на розгортання нової ВМ. Але запропонований механізм не враховує існуючі ВМ і обмеження на кількість міграцій ВМ з/на ФС. Інше обмеження полягає в тому, що при розміщенні ВМ не враховується час його виконання.

У роботі [180] запропонований багатокритеріальний алгоритм на основі мурашиної колонії з метою розв’язання задачі розміщення ВМ через отримання набору неприйнятих рішень, які одночасно зводять до мінімуму загальну витрату ресурсів і споживання енергії. Однак не враховано обмеження на кількість одночасних міграцій ВМ з/на кожний ФС і для обчислення рішень використовуються тільки ресурси ЦП і пам'яті, що споживають ВМ і ФС. Запропонований алгоритм розміщення ВМ, також, не дозволяє динамічно консолідувати існуючі ВМ.

У дисертаційній роботі підхід до вирішення проблеми консолідації ВМ базується на евристиці Power Aware Best Fit Decreasing (PABFD) [194], але, в той

же час, збалансоване навантаження кожного фізичного ресурсу ФС оптимізовано з використанням техніки імітаційного відпалу з урахуванням обмеження на максимальну кількість одночасних міграцій з/на кожний ФС.

Застосування алгоритму променевого пошуку

Останнім часом запропоновано багато методів та алгоритмів для вирішення проблеми управління ресурсами ЦОД [178, 107, 308]. Зокрема, задача консолідації ВМ розглядається як оптимізаційна задача з різними цільовими функціями. Також, задача консолідації ВМ розглядається як багатокритеріальна оптимізаційна задача [191, 201]. Складність цієї задачі полягає в наявності великої кількості станів середовища та обмежень. Крім того, складність виявляється при формулюванні цільової функції, до якої входять декілька показників, які треба оптимізувати. Унаслідок аналізу існуючих рішень з'ясувалося, що досягти ефективності деяких показників одночасно виявляється неможливим. Наприклад, неможливо досягнути одночасно високої швидкості розгортання ВМ та енергозбереження, або одночасно високої продуктивності та енергозбереження.

Для промислових ЦОД з хмарними інфраструктурами використовуються прості алгоритми управління, такі як first-fit, best-fit та їх модифікації (Eucalyptus [202], Microsoft [203], Google [204]). Це обумовлено вимогами робастності застосувань клієнтів та участю адміністраторів в автоматизованому процесі управління ресурсами ЦОД при постійному моніторингу якісних показників.

В дослідженнях використовуються такі цільові функції, як мінімізація споживання електроенергії, мінімізація порушень угод про якість сервісу SLA, мінімізація мережевого трафіку, максимізація продуктивності та використання ресурсів. Однією з основних умов для сучасних кластерів ФС є можливість роботи методів управління ресурсами в режимі онлайн [308]. Ці методи використовують потоки управління, що працюють *паралельно*. Разом з цим допускається застосування методів управління ресурсами, які спрацьовують при появі певних умов роботи кластеру, або через певний проміжок часу. Такі методи використовують потоки управління, що працюють *послідовно*. Для такого

випадку нові завдання зазвичай розміщуються на ФС, які не задіяні у процесі консолідації ВМ.

Крім традиційних евристик та детермінованих алгоритмів при управлінні ресурсами ЦОД використовуються і алгоритми локального пошуку, такі як еволюційні алгоритми [181], алгоритми оптимізації мурашиних колоній [180], табу пошук [187], пошук з емуляцією відпалу [182]. Більш ефективним на нашу думку є використання локального пошуку на другій стадії оптимізації, після підготовки відповідного набору станів детермінованими алгоритмами з метою покращити рішення, знайдене на першій стадії.

Таким чином, в дисертаційній роботі проаналізовано застосування алгоритму променевого пошуку в складі двостадійного методу для управління ресурсами хмарного ЦОД з метою зменшити кількість міграцій ВМ та збільшити кількість ФС, переключених в режим сну.

Аналіз існуючих рішень з управління хмарними ресурсами на основі алгоритму навчання з підкріпленням

У дослідженнях [215], [108], [216], [16] зроблений акцент на такі аспекти управління обчислювальними ресурсами ЦОД, як виділення, розподіл, перерозподіл, планування та забезпечення продуктивності. Відповідно, для розв’язання задачі розміщення ВМ запропоновано різні рішення з метою отримання оптимального плану розміщення ВМ за критерієм мінімізації кількості ФС та мінімізації кількості порушень угоди SLA.

Багато попередніх досліджень, що стосуються управління обчислювальними ресурсами ЦОД з багатокритеріальною оптимізацією [178, 220, 221], зосереджуються на трьох основних критеріях, таких як: забезпечення SLA між постачальником хмарних послуг та користувачем, зменшення енергоспоживання ЦОД, а також зменшення експлуатаційних витрат на управління послугами, що надає ЦОД. У огляді досліджень [108] проаналізовано різні підходи для пошуку оптимального розподілу ресурсів з метою мінімізації кількості ФС одночасно з виконанням умов SLA.

У роботі [194] автори детально проаналізували проблеми енергоефективності та ефективності динамічної консолідації ВМ. Автори

аналізують онлайн, офлайнову, детерміновану та динамічну проблеми консолідації ВМ та пропонують адаптивну евристику для динамічної консолідації ВМ. Недоліком запропонованих способів є те, що кількість одночасних міграцій з/на ФС не обмежена. Крім того, не враховується споживання енергії ФС в сплячому режимі та споживання енергії при переході з сплячого режиму в активний режим.

Слід також зазначити, що в деяких дослідженнях, таких як [222, 223, 224, 225], запропоновано алгоритми і методи, які оцінюються за допомогою спрощеного навантаження, тобто враховується тільки навантаження на ЦП, що взяті з трейсів моніторингу PlanetLab [15]. У той же час, такі ресурси, як оперативна пам'ять, підсистема збереження і мережеві інтерфейси не розглядаються в запропонованих моделях. Більше того, для всіх сценаріїв моделювання використовувались застарілі конфігурації ФС і ВМ.

У деяких попередніх дослідженнях [226, 227, 228, 229, 230] повідомляється, що час перемикання ФС з сплячого режиму в активний режим може досягати 200 секунд. Крім того, за час перемикання, енергоспоживання ФС досягає пікового значення, як і в активному режимі. Таким чином, перехідні режими роботи ФС в хмарних ЦОД істотно впливають на енергоспоживання і не повинні ігноруватися.

Одним з перспективних підходів до оптимального розподілу хмарних ресурсів між ВМ є навчання з підкріпленням (НП) [206]. У [208] управління віртуальними ресурсами в хмарі розглядається як проблема автоматичного керування з використанням підходу НП. Автори застосували методику Q-навчання [209] для управління кількістю ВМ, що забезпечують роботу хмарного застосунку. Запропонований алгоритм зберігає значення пар "дія-винагорода" як історичні значення і використовує їх для вирішення питання про зміну кількості ВМ під час роботи хмарного застосунку. Слід зазначити, що запропонований підхід орієнтований на довгострокову роботу застосунку і не враховує розміщення ВМ на ФС.

Онлайн гібридний алгоритм НП для динамічного розподілу ФС серед декількох веб-застосунків запропонований в [217]. При цьому поєднано НП з

моделями масового обслуговування в гібридному підході, в якому алгоритм НП налаштовується в автономному режимі на даних, зібраних у той час як політика моделей масового управління керує ресурсами ЦОД. Ефективність запропонованого алгоритму досліджено на спрощеному прототипі ЦОД. Але запропонований алгоритм обмежений орієнтацією на веб-застосунки і не може бути застосований до віртуальних середовищ.

Щоб отримати баланс між прибутком від забезпечення QoS і витратами на споживання енергії для забезпечення заданої потужності, в [225] запропоновано алгоритм адаптивного управління ресурсами на основі НП. Запропонований алгоритм не вимагає попереднього знання вимог до ресурсів і є надійним при фактичному навантаженні. Метою запропонованого підходу є пошук оптимального балансу між доходом від забезпечення QoS та витратами на електроенергію. Але в запропонованому підході враховується лише використання ЦП і не розглядаються інші ресурси ФС. Крім того, запропонований алгоритм не враховує вплив змін оперативної пам'яті ВМ на накладні витрати при міграції ВМ.

У роботі [231] вирішується задача динамічного ресурсного забезпечення ФС за допомогою алгоритму навчання на основі апостеріорного рішення з швидкою конвергенцією. Досліджено вплив розміру простору станів на продуктивність запропонованого підходу. У результаті зроблено висновок, що збільшення розміру простору станів зменшує швидкість збіжності пошуку рішення. Введені, також, деякі нові методики, що забезпечили прискорення конвергенції запропонованого алгоритму порівняно з традиційним алгоритмом Q-навчання. Однак запропонований підхід враховує тільки гомогенні ФС. Крім того, використовується модель розподілу робочого навантаження, яка не може бути точно визначена у виробничому середовищі зі змішаним навантаженням.

Децентралізована архітектура енергозберігаючої системи управління ресурсами для ЦОД представлена у роботі [207]. Визначені проблеми мінімізації споживання енергії при виконання вимог QoS та сформульовані вимоги для політики розподілу ВМ. Запропоновано три стадії безперервної оптимізації розміщення ВМ: перерозподіл відповідно до поточного споживання декількох

системних ресурсів, оптимізації віртуальних мережевих топологій, встановлених між ВМ та перерозподіл ВМ з урахуванням теплового стану ресурсів. Для першої стадії застосовані евристичні алгоритми, які були оцінені за допомогою середовища моделювання CloudSim. Запропонований алгоритм мінімізації міграцій дозволив значно зменшити споживання енергії в ЦОД.

У роботі [210] представлено метод уніфікованого навчання з підкріпленням, що відкриває новий напрям в автоконфігуруванні ВМ та апаратних засобів в умовах хмарних обчислень. У цьому методі агент НП регулює конфігурацію ВМ щоб максимізувати свою винагороду в довгостроковій перспективі. Для прискорення процесу навчання в великомасштабних системах розроблено різні моделі апроксимації винагород від дій по конфігурації ВМ. Експериментальні результати з різним робочим навантаженням продемонстрували ефективність методу.

Щоб мінімізувати кількість увімкнутих ФС з урахуванням вимог поточних навантажень, в [218] запропоновано метод динамічної консолідації ВМ на основі НП, що використовує агента, який обчислює оптимальну політику для визначення режиму роботи ФС (очікування або активний). Агент навчається приймати рішення вибору режиму роботи ФС на основі зібраних даних і покращує свої рішення при зміні робочого навантаження. Запропонований метод не вимагає створення моделі ЦОД і динамічно пристосовується до змін у середовищі. Але запропонований метод враховує тільки навантаження на ЦП і не враховує інші ресурси ФС. Крім того, запропонований алгоритм не враховує витрати на міграції ВМ.

З метою призначення відповідних ресурсів для ВМ з визначеними вимогами до продуктивності, в [219] запропоновано підхід VCONF, що базується на НП. VCONF дозволяє автоматизувати процес вертикального нарощування ресурсів ВМ у відповідь на зміну вимог з боку застосунків і навантаження. Запропонований підхід використовує алгоритми НП на основі моделі ЦОД для уникнення проблем масштабованості та адаптивності алгоритмів НП, що застосовуються в реальних умовах управління ресурсами ЦОД. Він може знайти оптимальні конфігурації ВМ в системах малого масштабу і демонструє хорошу

адаптивність і масштабованість. Але запропонований підхід не враховує необхідність міграції ВМ при відсутності фізичних ресурсів на поточному ФС.

В публікаціях [205, 211] розглянуто підхід до управління ресурсами ЦОД, призначений забезпечити ефективне їх використання у процесі функціонування ЦОД за схемою виділених серверів. Наводяться моделі і алгоритми розподілу ресурсів, структура відповідних інструментальних засобів. Описано три варіанти системи, яка надає хостингові послуги на основі моделі виділених серверів: виділені сервери на базі сервіс-орієнтованої архітектури, виділені сервери на базі традиційної трирівневої архітектури та гібридна архітектура.

Для моделі планування при надлишку ресурсів наведені чотири задачі мінімізації витрат на підтримку частини серверів, яка забезпечуватиме підтримку запитів користувачів клієнта. Для розв’язання наведених вище задач оптимального розміщення екземплярів застосувань пропонуються варіанти генетичного алгоритму (ГА) і евристичного алгоритму, який враховує специфіку критеріїв і обмежень. Крім того, пропонується комбінований алгоритм, побудований на використанні методів вирішення задач лінійного програмування і неявного перебору.

У роботі [212], на основі узагальнення накопиченого досвіду розв’язання проблеми розподілу і управління навантаженням і ресурсами центрів оброблення даних, пропонуються модифікації раніше розроблених моделей. З метою розроблення універсального алгоритму для цього класу задач авторами пропонується варіант керованого ГА. Керованість означає вплив на вибір чергових операторів формування популяції у залежності від деяких важливих її параметрів. Через налаштування зазначених параметрів керований ГА можна досить швидко навчити розв’язувати різноманітні проблеми зазначеного класу, особливістю яких є складний характер взаємодії критеріїв і обмежень, що раніше часто призводило до передчасного виродження популяції. Експериментальне дослідження показало, що керований ГА дозволяє отримати кращі результати відносно базового ГА.

1.5. Вимоги до застосування моделей і методів прогнозування при управлінні ІТ-інфраструктурою хмарного центру оброблення даних

Управління віртуалізованими ресурсами з використанням прогнозування широко використовується в системах управління провайдерів хмарних послуг і висвітлюється в публікаціях [2, 3, 4, 5, 6, 7, 8, 9, 93, 94, 95, 353]. Більшість публікацій присвячена прогнозуванню потреби в ресурсах центрального процесора, так як вважається, що нестача цього ресурсу найбільше впливає на якість надання хмарних послуг. Зокрема, у працях [10, 11, 12] наведено методи, що використовуються виключно для прогнозування навантаження процесора в реальному часі та проаналізовано різні моделі прогнозу та алгоритми навчання нейронних мереж.

У роботі [6] запропонований фреймворк для прогнозування навантаження PRACTISE, що базується на використанні нейронної мережі. Зроблений порівняльний аналіз фреймворку з методом авторегресії інтегрованого ковзного середнього (ARIMA) [13] та звичайною нейронною мережею. Автори зібрали статистику споживання ресурсів декількох тисяч ВМ в період двох місяців з інтервалом 15 хвилин. Але запропонований алгоритм надає прогноз тільки на один день, що не може бути використано при оперативному управлінні ВМ, наприклад, впродовж години. Обсяг навчальної вибірки при цьому обраний довільно і фіксовано. Інші дані, які можуть бути отримані під час роботи в реальних умовах, не враховуються. Повторне тренування моделі відбувається у відповідь на появу сплесків помилки прогнозування. Але не вказано, скільки вимірів взято для повторного тренування моделі, і яким чином формуються різні розбиття. Крім того, опис моделей ARIMA та звичайної нейронної мережі, навчальні вибірки, а також деталі їх отримання автори не наводять.

Автори роботи [7] представили метод прогнозування споживання ресурсів процесора, що базується на лінійній регресії [14]. Запропонований метод дає можливість отримати короткостроковий прогноз. Для перевірки працездатності автори використали набір даних PlanetLab [15], що містить статистику споживання процесорного ресурсу ВМ з інтервалом вимірювань 5 хвилин. Але в роботі автори не врахували, що процес вхідних і вихідних міграцій на ФС, а

також різна кількість і склад ВМ, значно впливають на споживання ресурсу процесора ФС. Тому, після завершення міграцій необхідно знов будувати моделі прогнозу. Крім того, автори використовують фіксований розмір навчальної вибірки довжиною 12 вимірів для побудови моделі прогнозу, що не може бути виправданим для всіх сценаріїв споживання процесорного ресурсу в промислових умовах.

У роботі [8] автори розглядають проблему прогнозування споживання процесорного ресурсу як задачу аналізу часових рядів. Запропоновано прогнозуючий фреймворк, який використовує статистичні моделі прогнозу споживання ресурсу процесора з метою завчасного масштабування фізичних ресурсів і запобігання проблем недостатнього їх виділення. Запропонований фреймворк базується на використанні нейронної мережі та лінійної регресії у поєднанні з технікою рухомого вікна (sliding window technique). Але для тренування моделі автори використовують штучні набори даних, отримані генератором навантажень TPC-W. Крім того, кількість вимірів навантаження в наборі відносно мала і складає 135 вимірів для тренування і перевірки моделей. Таким чином, отримана модель не може бути використана для прогнозування потреби в процесорних та інших ресурсах в промислових умовах. Як саме була побудована модель прогнозу з використанням рухомого вікна автори не уточнюють, але залежність помилки прогнозу від розміру вікна, наведена авторами, показує, що найкраща якісь прогнозу досягається різним розміром вікна для різних використаних методів. Цей факт не дає можливість визначити конкретний розмір рухомого вікна при прогнозуванні потреби в ресурсі.

Адаптивний енергоефективний фреймворк для забезпечення фізичних ресурсів застосуванням хмарного ЦОД запропонований у роботі [9]. Він включає в себе кластеризацію типів навантаження та прогнозування кількості запитів для певного класу ВМ за допомогою фільтра Вінера. Для перевірки працездатності та оцінки роботи запропонованого фреймворку використано трейси Google. Кожної хвилини фреймворк вираховує нові коефіцієнти моделей прогнозу для кожного типу навантаження, адаптуючись до поточних умов роботи ВМ. Але отримані моделі прогнозу виявились налаштованими таким чином, що фізичні

ресурси, згідно прогнозу, виділяються із значним запасом. Це дозволяє запобігти порушенню SLA, але призводить до надмірного виділення ресурсів і збільшення енергоспоживання. Крім того, не враховується вплив міграцій ВМ на споживання обчислювальних ресурсів.

Багато методів прогнозування, що використовують комбінування прогнозів, представлені у працях [96, 97, 98, 99, 100]. Аналіз попередніх досліджень показує, що комбінація моделей прогнозування зазвичай перевершує індивідуальні прогнози, отримані окремими альтернативними методами або моделями. Крім того, дослідження автора [286, 327] показують, що для нестаціонарного часового ряду моделі прогнозу, що дозволяють отримати гарний прогноз, важко ідентифікувати.

Багато зусиль для покращення точності прогнозування через комбінування моделей направлені на розроблення однорідних і гетерогенних комбінованих методів прогнозування. Запропоновані методики можна розділити на дві групи:

- методи з різними алгоритмами прогнозування [100], такі як авторегресивні моделі (AR), моделі ARIMA, моделі простого експоненціального згладжування (SES), моделі векторної регресії (SVR), штучні нейронні мережі (ANN), метод групового врахування аргументів [101] і т.д.;
- методи, що мають подібну структуру моделі, але використовують різні критерії вибору змінної [99, 98].

Таким чином, основний акцент пропонованих методів полягає в прогнозуванні часових рядів для процесу прийняття рішень. Але, комбінування прогнозів для управління ресурсами хмарного ЦОД в реальному часі не реалізовано на даний момент. Незважаючи на існуючі підходи застосування комбінування моделей прогнозування, комбінація прогнозів для управління в реальному часі все ще залишається недостатньо розвиненою.

Використані в дисертації альтернативні моделі і методи прогнозування

Просте експоненційне згладжування

Метод простого експоненційного згладжування (simple exponential smoothing, SES) використовується для прогнозування часових рядів у випадках,

якщо дані спостереження не мають вираженого тренду і сезонності, або ці характеристики можуть з'являтися в даних тимчасово [19].

При обчисленні прогнозу ступінь врахування попередніх значень ряду зменшується за експоненційним законом (1.1).

$$\hat{y}_{t+1|t} = \alpha y_t + \alpha(1-\alpha) y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + \cdots, \tag{1.1}$$

де $0 < \alpha < 1$ – параметр згладжування, $\hat{y}_{t+1|t}$ – прогнозоване значення на один крок вперед в момент часу t, $y_t, y_{t-1}, y_{t-2}, \ldots$ – значення часового ряду, що входять до навчальної вибірки.

Метод Хольта та демпфірований метод тренду

Тренд Хольта (Holt's Linear Trend) та демпфірований метод тренду (Damped Trend) є методами, що розширюють можливості методу SES. Якщо в даних спостереження може бути наявний тренд ці методи можуть дати якісний прогноз [20].

Рівняння прогнозу методом Хольта представлено рівняннями (1.2).

$$\begin{gathered} \hat{y}_{t+1|t} = \ell_t + b_t \\ \ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta) b_{t-1} \end{gathered} \tag{1.2}$$

де ℓ_t – оцінка рівня часового ряду в момент часу t, b_t – оцінка тренду часового ряду в момент часу t, α, $0 \le \alpha \le 1$ – параметр згладжування для рівня часового ряду, β, $0 \le \beta \le 1$ – параметр згладжування для тренду часового ряду. Параметри згладжування і початкові значення ℓ_t та b_t для моделі прогнозу оцінюються через мінімізацію суми квадратичних похибок.

Емпіричні дослідження часових рядів показали, що метод Хольта дає "надмірний" прогноз. Значно кращі результати дає демпфірування тренду Хольта через введення спеціального параметру демпфірування [21].

Рівняння одно-крокового прогнозу методом демпфіруваного тренду Хольта представлене рівняннями (1.3).

$$\hat{y}_{t+1|t} = \ell_t + \phi b_t$$

$$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \phi b_{t-1}) \quad (1.3)$$
$$b_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)\phi b_{t-1}$$

Для коефіцієнта ϕ зазвичай встановлюють значення в діапазоні [0.8 – 0.98].

Модель авторегресії з інтегрованим ковзним середнім (ARIMA)

Модель авторегресії з ковзним середнім (ARMA) представлена рівнянням (1.4).

$$y_t = \phi_0 + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t, \quad (1.4)$$

де ϕ_0 і ϕ_i – коефіцієнти моделі авторегресії порядку p, θ_i – коефіцієнти моделі ковзного середнього порядку q, ε_t – білий шум ($E(\varepsilon_t) = 0, Var(\varepsilon_t) = \sigma_\varepsilon^2, Cov(\varepsilon_t, \varepsilon_s) = 0, t \neq s$).

Для прогнозу наступного значення часового ряду за допомогою моделі ARMA використовується вираз (1.5) із застосуванням оператора зсуву.

$$\hat{y}_{t+1|t} = \sum_{i=0}^{p-1} \phi_i L^i y_t + \sum_{i=0}^{q-1} \theta_i L^i \varepsilon_t + \varepsilon_{t+1}, \quad (1.5)$$

де L^i – оператор зсуву, визначений як $L^i y_t = y_{t-t}$, ϕ_i і θ_i – коефіцієнти моделі, ε_t – білий шум.

Якщо часовий рад, отриманий від підсистеми моніторингу, є нестаціонарним, для прогнозу треба отримати модель ARIMA [13] порядку d через взяття перших (d=1) або других (d=2) різниць між членами ряду. Зазвичай, щоб отримати стаціонарний ряд і скористатися моделлю авторегресії ковзного середнього, на практиці вистачає взяти перші різниці членів ряду. Таким чином, модель ARIMA(p,d,q) представлена рівнянням (1.6).

$$(1 - \sum_{i=0}^{p-1} \phi_i L^i)(1-L)^d y_t = (1 + \sum_{i=0}^{q-1} \theta_i L^i)\varepsilon_t \quad (1.6)$$

Модель лінійної регресії з трендом

Ще одним альтернативним методом, який може бути використаний для прогнозування значень навантаження є лінійна регресія. Дані спостережень часових рядів на деяких їх ділянках можуть мати тренди і сезонність. У цьому

випадку підбирається лінійна модель до часового ряду і з використанням предиктора обчислюється прогнозне значення. При цьому використовується предиктор у вигляді тренда:

$$y_t = b_0 + b_1 t + \varepsilon_t,$$

де $t = [1,..., w_{\max}]$ – кількість відліків часу, для яких значення функції y_t взяті для побудови моделі, $w_{\max}$ – максимальний розмір навчальної вибірки для створення моделі.

Унаслідок вирішення рівняння $\mathbf{B} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y}$, де $\mathbf{B} = (b_0, b_1)^T$ – вектор параметрів лінійної моделі, $\mathbf{Y} = (y_1, y_2, \ldots, y_{w_{\max}})$ – вектор значень часового ряду, $\mathbf{A}^T = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ t_1 & t_2 & \cdots & t_{w_{\max}} \end{pmatrix}$ – матриця рівняння, обчислюються параметри моделі. Отримана модель використовується для прогнозування значення споживання ресурсу на наступному кроці.

Метод TBATS

Метод TBATS (Trigonometric Box-Cox transform, ARMA errors, Trend, and Seasonal components), запропонований в роботі [28], використовує комбінацію методів і перетворень, що базуються на розкладі в ряд Фур'є, моделі станів експоненційного згладжування і Box-Cox трансформації. Цей метод дає можливість отримати достатньо адекватні моделі для часових рядів, в яких сезонність може змінюватись. Оскільки в цьому методі деякі параметри підбираються автоматично, адекватність отриманої моделі може бути низькою. Крім того, в роботі [28] зазначено, що час виконання цього методу може бути порівняно великим.

1.6. Особливості моделей, методів і технологій управління IT-інфраструктурою систем інтернету речей

Інтернет речей являє собою розгалужену мережу об'єктів, що мають засоби комунікації і обчислювальні потужності з метою виконання певних дій в оточуючому середовищі. IoT дозволяє впроваджувати та покращувати роботу таких важливих сервісів, як адміністрування міських (муніципальних) сервісів,

закладів охорони здоров’я та освіти, управління міським транспортом та системами життєдіяльності. Успішне впровадження систем IoT значно залежить від якості сервісів, що надаються. Таким чином, актуальною є науково-практична проблема розроблення технологій і підходів до управління IT-інфраструктурою IoT систем [279].

Згідно проекту IEEE P2413 [175] в архітектурі IoT виділяють три основних рівня: рівень застосувань та сервісів (Applications), мережевий та комунікаційний рівень (Networking and Data communication), та рівень давачів (Sensing). Кожний рівень представлений сукупністю пристроїв та програмних засобів, а також протоколів, що забезпечують їх взаємодію крізь рівні з метою надання певного сервісу в рамках окремого домену (охорона здоров’я, транспорт, промисловість, страхування, послуги та ін.). Розглянемо технології та сучасні рішення для IT-інфраструктури IoT системи, що забезпечує функціонування рівня застосунків та сервісів, а також комунікаційного рівня. При цьому IT-інфраструктура є основою розроблення та впровадження технологій, на яких базується робота IoT системи.

Останнім часом спостерігається зростання бізнес-попиту на існуючі та нові IoT сервіси, що надають IoT системи. Таким чином, з’являється необхідність розроблення та впровадження нових підходів до управління IT-інфраструктурою IoT систем.

Сучасні системи управління IT-інфраструктурою представляють собою складні системи, що інтегрують рішення і технології від різних виробників. Зростаюча складність СУІ супроводжується зростанням вартості обслуговування IT-інфраструктури. Основна задача системи управління IoT інфраструктурою підтримувати належний рівень IT сервісу за рахунок використання необхідної кількості ресурсів IT-інфраструктури в умовах віртуальних, кластерних та розподілених середовищ при змінній інтенсивності запитів користувачів. IoT сервіси надають нові можливості у всіх сферах життєдіяльності людини, але створюють додаткові навантаження на ЦОД за рахунок постійного додавання все нових і нових пристроїв у мережі систем IoT, що призводить до зростання обсягів обміну даними та їх оброблення [279].

Досвід розгортання систем IoT [176] показав, що їм притаманна висока складність, яку не можливо порівняти з розгортанням традиційних IT систем і застосувань, додаванням груп комп'ютерів, мобільних пристроїв та сенсорів до мережі. Щоб врахувати всі особливості і проблеми при розробленні та впровадженні нових проектів IoT підприємства повинні розробити та впровадити відповідні технологічні зміни щоб існуюча IT-інфраструктура була готова до зростання навантажень при обміні та обробці великих обсягів даних.

Європейський дослідницький кластер з питань IoT (IoT European Research Cluster, IERC) визначає інтернет речей у такий спосіб. IoT є динамічною глобальною мережевою інфраструктурою з властивостями самоконфігурування, що базується на стандартних протоколах. У цій інфраструктурі, фізичні та віртуальні пристрої ідентифікуються та визначаються в певному просторі імен, мають відповідні атрибути та властивості, використовують інтелектуальні інтерфейси та безшовно інтегруються в інформаційну мережу [149].

Багато нагальних IoT застосувань розробляються та розгортаються із застосуванням хмарних технологій [177]. Відповідно існує необхідність в розробленні та застосуванні нових технологій оброблення і зберігання даних в ЦОД щоб забезпечити продуктивність, надійність та еластичність.

Значні наукові та практичні результати отримані в галузі розроблення розумних (англ. Smart) систем для певних доменів та територіальних одиниць, наприклад, розумний дім (Smart Home), розумне місто (Smart City), розумний офіс (Smart Office) та ін. Еволюція технологій та нових рішень в цьому напрямку відбувається дуже стрімко [150]. Найбільші темпи розвитку треба відмітити в екосистемі машина-машина (Machine-to-Machine, M2M), яка потребує оброблення великих обсягів даних, отриманих з сенсорів [151]. Дані, отримані з давачів, необхідно передавати до центрів оброблення у відповідні застосування у відповідності з політиками безпеки та у визначені часові періоди. Хмарні IoT сервіси використовують збережені дані для аналітичної обробки з метою прийняття якісних рішень при управлінні бізнес процесами. Згідно прогнозів, кількість IoT пристроїв збільшиться і досягне 212 мільярдів до 2020 року. Передача даних в мережі Internet, що стосуються роботи IoT систем, досягне

частки 45% від всього трафіку [152]. Збільшення кількості пристроїв в системах IoT впливає на конфігурацію ЦОД. Гартнер передбачає до 2020 року підключення до мережі Інтернет 25 мільярдів пристроїв, що створить додаткове навантаження на комунікаційні канали та системи зберігання даних ЦОД. У своїх документах Європейський інститут стандартів телекомунікацій (European Telecommunications Standards Institute, ETSI) акцентує увагу саме на терміні M2M, а не інтернет речей. ETSI визначає M2M як взаємодію сутностей, при якій немає необхідності втручання людини. При цьому, кінцева мета послуг M2M – автоматизувати процеси прийняття рішень та обміну інформацією [153]. Згідно дослідженням Forrester [154] розумне середовище (a smart environment) використовує ІКТ щоб додати інтерактивності і ефективності компонентам критичних інфраструктур, службам міського адміністрування, компонентам в складі систем управління процесами в освіті, охороні здоров'я, забезпеченні безпеки, в громадському транспорті та ін.

Останнім часом в різних публічних хмарних середовищах розгорнуто багато платформ IoT, що дозволяє порівнювати їх за різними показниками, такими як доступність, відмовостійкість та еластичність. Крім того, актуальною є задача розроблення і дослідження нових підходів і методів з метою покращення роботи хмарних IoT сервісів. У статті [170] представлені підходи до організації та інтеграції інфраструктур і сервісів IoT в хмарному середовищі. Автори [171] проаналізували шляхи і засоби інтеграції IoT з хмарними середовищами, а також запропонували парадигму хмарного IoT. Потенційні можливості технологій IoT і хмарних технологій для задоволення потреб бізнесу у сфері виробництва та управління промисловими процесами досліджені в роботі [172]. У цій статті автори запропонували інтегровану архітектуру хмарну систему виробничих сервісів, що поєднує в собі хмарні сервіси та технології IoT на виробництві. Ключові виклики на шляху інтеграції технологій хмарних обчислень та IoT, що полягають у зростанні кількості пристроїв IoT, інтеграції гетерогенних пристроїв, обмеження на споживання енергії та пропускну здатність каналів зв'язку проаналізовані в [173].

Багато застосунків IoT не повинні залежати від особливостей місцезнаходження, потребують малих затримок при передачі даних та мобільності. При цьому, дані повинні оброблятися в реальному часі. Це стосується даних, що впливають на прийняття оперативних рішень в середовищах з різними характеристиками подій в часі (транспортні рухи, паркування автомобілів, логістичні сервіси та ін.). У цих умовах актуальною проблемою є розроблення і застосування концепції мікро хмари (Micro cloud, fog computing), надають можливість створювати нові застосунки управління даними та аналітики з метою розширити парадигму хмарних обчислень та технологію мереж доставки контенту (Content Delivery Network) на периферію мережі, ближче до споживачів та джерел інформації. Це дасть змогу наблизити відповідну частку ресурсів (обчислення, сховище, застосування, сервіси) до конкретних територій та кінцевих користувачів [155].

Потужність та кількість IoT сервісів, що надаються користувачам крупними провайдерами послуг, швидко зростає. Багато з цих сервісів працюють без зупинки взагалі. Це також призводить до накопичення, обробки і перерозподілення великих обсягів даних [279].

Таким чином, при створенні систем управління інфраструктурою IoT на базі мікро хмар для розгортання відповідних сервісів і застосунків необхідно забезпечити достатній рівень еластичності та належні характеристики взаємодії мікро хмар (або кластерів) із застосуванням відповідної мережі. З цією метою, для реалізації мікро хмар, пропонується використовувати гіперконвергентні системи [174], що забезпечують належну еластичність при зростанні вимог до ресурсів з боку застосунків IoT.

1.7. Формулювання науково-технічної проблеми та завдань дослідження

Результати аналізу хмарного ЦОД як об’єкту керування показали, що інформаційна технологія управління IT-інфраструктурою таких об’єктів повинна розроблятися з урахуванням ієрархії інформаційних процесів, які в цих умовах розглядаються як хмарні послуги, з урахуванням наявності віртуалізованих середовищ на нижньому рівні ієрархії хмарних послуг, і з

урахуванням включення програмно-визначених систем в ланки управління хмарними послугами на всіх рівнях ієрархії.

В п. 1.1 проаналізовано стан хмарних ЦОД і процесів управління його ресурсами з урахуванням сервісних моделей хмарних обчислень, новітніх апаратно-програмних засобів побудови хмарної інфраструктури на базі ГІ і СЗД з метою реалізації інформаційних процесів і надання їх у вигляді хмарних послуг. Унаслідок аналізу виникає необхідність розв’язання задач управління ресурсами ІТ-інфраструктури провайдера хмарних послуг із застосуванням новітніх методології, підходів, моделей і методів, що базуються на методах інтегрованого і ієрархічного управління, методах штучного інтелекту, методах прогнозування навантажень і споживання ресурсів ІТ-інфраструктури, проаналізованих в пп. 1.2, 1.3, 1.4. При цьому, якщо згідно парадигми хмарних обчислень [60], з боку споживача ресурси хмарного ЦОД виглядають необмеженими, то з боку провайдера хмарних послуг існують суттєві обмеження на кількість апаратних ресурсів, енергоспоживання, дотримання умов угоди про рівень обслуговування, капітальні та операційні витрати, кількість збоїв та простоїв та ін.

Вказані обставини обумовлюють необхідність раціонального використання ресурсів хмарного ЦОД провайдера, а також забезпечення заданої продуктивності надання послуг і їх безпеки. Тому розроблення інформаційної технології управління ІТ-інфраструктурою ЦОД провайдера хмарних послуг є актуальною науково-технічною проблемою.

Таким чином, при розробленні інформаційної технології управління ІТ-інфраструктурою хмарного ЦОД в умовах невизначеності і змінних навантажень виникають такі найбільш важливі для розв’язання задачі:

- оптимізація розподілу ресурсів між споживачами хмарних послуг;
- забезпечення якості хмарних послуг у відповідності із заданими в угоді про рівень обслуговування;
- оптимізація споживання електроенергії при забезпеченні заданої продуктивності роботи хмарних сервісів.

Сучасний стан розв’язання вказаних задач розглянутий і проаналізований в пп. 1.1 – 1.5. Однак наявні підходи до розв’язання вказаних задач не відповідають сучасним вимогам до якості систем управління ІТ-інфраструктурою хмарних ЦОД і не враховують особливостей, вимог, сервісних моделей і характеристик хмарних обчислень [60]. По-перше, вказані вимоги і характеристики пов’язані з різноманітністю інформаційних процесів і їх еволюцією, що призводить до необхідності покращення роботи існуючих систем управління ІТ-інфраструктурою. По-друге, ці вимоги пов’язані з обмеженістю доступу провайдерів хмарних послуг в різних країнах до сучасних технологій управління ресурсами ІТ-інфраструктури, що закриті для доступу глобальними корпораціями-розробниками хмарних технологій для ЦОД.

Розроблення інформаційної технології управління ІТ-інфраструктурою хмарного ЦОД передбачає розв’язання всіх задач управління і керування на всіх рівнях архітектури в їх взаємозв’язку. Для розроблення інформаційної технології на основі ієрархічного, адаптивного і декомпозиційно-компенсаційного підходів необхідно вирішити такі задачі:

1) аналіз існуючих підходів, технологій, моделей і методів управління ІТ-інфраструктурою ЦОД провайдерів хмарних послуг, уточнення проблеми і задач дослідження;
2) розроблення методології управління ІТ-інфраструктурою хмарного ЦОД як систему новітніх підходів, моделей і методів з використанням стратегій управління і схем їх реалізації для відповідних умов функціонування;
3) розроблення моделей і методів прогнозування змішаних навантажень на ресурси ІТ-інфраструктури хмарного ЦОД в умовах невизначеності з використанням принципів адаптації та комбінування;
4) розроблення комплексу моделей і методів інтегрованого управління ресурсами, потужністю і сховищем ІТ-інфраструктури хмарного ЦОД на базі алгоритмів і методів стохастичного локального пошуку з використанням прогнозування змінних навантажень, нових метрик оцінювання стану та дотриманням вимог угоди про рівень обслуговування;

5) подальший розвиток алгоритмів і методів стохастичного локального пошуку з урахуванням особливостей управління ресурсами і навантаженням ІТ-інфраструктури хмарного ЦОД в умовах невизначеності;
6) розроблення архітектури багаторівневої програмно-визначеної системи управління ІТ-інфраструктурою хмарного ЦОД і концепції інформаційної технології на базі підходів і методів інтегрованого та ієрархічного управління;
7) подальший розвиток декомпозиційно-компенсаційного підходу до управління хмарним ЦОД з урахуванням адаптивності, багаторівневості та програмно-визначених підходів його функціонування;
8) розроблення і реалізація інформаційної технології управління ІТ-інфраструктурою хмарного ЦОД та експериментальне дослідження її ефективності, а також розроблених моделей, алгоритмів і методів.

Висновки до розділу 1

1. Дисертаційна робота виконана за спеціальністю 05.13.06 – інформаційні технології. Із формули спеціальності використано: розроблення принципів оптимізації та моделей і методів прийняття рішень за умов невизначеності при створенні автоматизованих систем різноманітного призначення, розроблення теоретичних і прикладних засад побудови і впровадження інтелектуальних інформаційних технологій для створення новітніх систем накопичування, переробки, збереження інформації та систем управління.

Із напрямів досліджень використано:

- розроблення наукових і методологічних основ створення і застосування інформаційних технологій та інформаційних систем для автоматизованої переробки інформації і управління;
- розроблення моделей і методів автоматизації виконання функцій та завдань виробничого і організаційного управління в звичайних і багаторівневих структурах на основі створення та використання нових інформаційних технологій;

- моделювання предметних галузей інформаційних систем (аналітичне, імітаційне, інфологічне об’єктно-орієнтоване, тощо) на підґрунті створення і застосування відповідних інформаційних технологій;
- розроблення теоретичних і прикладних основ побудови інформаційних технологій для автоматизації функціональних завдань керування аналізу і оцінювання ефективності автоматизованих систем переробки інформації та управління.

2. Аналіз праць вітчизняних і зарубіжних авторів показав, що ІТ-інфраструктура хмарних ЦОД знаходиться в стадії інтенсивного розвитку і модернізації, що потребує покращення роботи існуючих систем управління ІТ-інфраструктурою і впровадження нових підходів і методів для підвищення ефективності систем управління ІТ-інфраструктурою з метою забезпечення заданої якості надання послуг при раціональному використанні ресурсів [349]. Сучасні інформаційні процеси реалізовані на множині елементів ІТ-інфраструктури. Складність організації процесів взаємодії елементів ІТ-інфраструктури не дозволяє створити їх адекватні моделі без додаткових обмежень, інструментальних засобів і галузевих стандартів. Системні властивості елементів ІТ-інфраструктури не досліджені належним чином.

3. Аналіз основних концепцій і підходів до управління ІТ-інфраструктурою хмарного ЦОД дозволив зробити такі висновки:

- сучасний ЦОД, що реалізує концепцію хмарних обчислень, характеризується складністю, ієрархічністю, багатовимірністю і багатоаспектністю;
- існуючі підходи і методи управління ІТ-інфраструктурою хмарного ЦОД зачасту не передбачають врахування взаємозв’язків і взаємного впливу на всіх рівнях ієрархії побудови інформаційних процесів, не адаптуються до інтенсивності використання та вивільнення ресурсів, а орієнтуються лише на розв’язання певної задачі на певному рівні (інфраструктури, платформи або ПЗ) [349];
- недостатньо уваги приділяється розробленню методів і моделей управління ІТ-інфраструктурою з використанням прогнозів споживання

обчислювальних ресурсів і прогнозів навантаження на елементи інфраструктури, а використані методи і моделі прогнозу орієнтовані на певну комбінацію навантажень і не адаптуються до поточних умов роботи ЦОД [349];

- недостатньо уваги приділяється розробленню інтегрованих методів управління ІТ-інфраструктурою, які б враховували одночасно аспекти віртуалізації, програмно-визначених систем, сховищ даних і гіперконвергентних систем.

4. Для вирішення виявлених проблем з метою забезпечення ефективного функціонування ІТ-інфраструктури ЦОД провайдера хмарних послуг запропоновано використовувати теорію систем, методи теорії ієрархічних систем, методи математичного програмування, методи дослідження операцій і теорії прийняття рішень, методи математичного та імітаційного моделювання, методи теорії штучного інтелекту, стохастичні і евристичні методи пошуку, методи прогнозування, методи математичної статистики, сервісні моделі хмарних обчислень для розроблення інформаційної технології управління за такими напрямками:

- аналіз існуючих підходів, технологій, моделей і методів управління ІТ-інфраструктурою ЦОД провайдерів хмарних послуг, уточнення проблеми і задач дослідження;
- розроблення методології управління ІТ-інфраструктурою хмарного ЦОД як систему новітніх підходів, моделей і методів з використанням стратегій управління і схем їх реалізації для відповідних умов функціонування;
- розроблення моделей і методів прогнозування змішаних навантажень на ресурси ІТ-інфраструктури хмарного ЦОД в умовах невизначеності з використанням принципів адаптації та комбінування;
- розроблення комплексу моделей і методів інтегрованого управління ресурсами, потужністю і сховищем ІТ-інфраструктури хмарного ЦОД на базі алгоритмів і методів стохастичного локального пошуку з використанням прогнозування змінних навантажень, нових метрик оцінювання стану та дотриманням вимог угоди про рівень обслуговування;

- подальший розвиток алгоритмів і методів стохастичного локального пошуку з урахуванням особливостей управління ресурсами і навантаженням ІТ-інфраструктури хмарного ЦОД в умовах невизначеності;
- розроблення архітектури багаторівневої програмно-визначеної системи управління ІТ-інфраструктурою хмарного ЦОД і концепції інформаційної технології на базі підходів і методів інтегрованого та ієрархічного управління;
- подальший розвиток декомпозиційно-компенсаційного підходу до управління хмарним ЦОД з урахуванням адаптивності, багаторівневості та програмно-визначених підходів його функціонування;
- розроблення і реалізація інформаційної технології управління ІТ-інфраструктурою хмарного ЦОД та експериментальне дослідження її ефективності, а також розроблених моделей, алгоритмів і методів.

РОЗДІЛ 2. КОНЦЕПТУАЛЬНІ ОСНОВИ УПРАВЛІННЯ ІТ-ІНФРАСТРУКТУРОЮ ХМАРНИХ ЦЕНТРІВ ОБРОБЛЕННЯ ДАНИХ

У розділі обґрунтовано необхідність врахування суттєвих характеристик хмарних обчислень при розробленні ІТУ ІТ-інфраструктурою хмарного ЦОД; розроблено методологію управління ІТ-інфраструктурою хмарного ЦОД на основі операторної форми постановки, аналізу і розв'язання задач управління в умовах невизначеності і змінних навантажень; операторний підхід до управління ІТ-інфраструктурою хмарного ЦОД; узагальнену схему реалізації функцій управління ІТ-інфраструктурою хмарного ЦОД; запропоновано розглядати ІТ-інфраструктуру хмарного ЦОД як нелінійний нестаціонарний дискретний об'єкт зі змінною структурою (ННДОЗС); розроблено схему реалізації операторної форми управління ІТ-інфраструктурою; розроблено структурно-функціональну модель СУІ хмарного ЦОД; розроблено задачі і стадії управління ІТ-інфраструктурою хмарного ЦОД з урахуванням ієрархії підсистем і стратегій управління; розроблені стратегії управління і моделі вибору стратегій при управлінні ресурсами хмарного ЦОД; обґрунтовано необхідність використання інтегрованих рішень і гіперконвергентних систем при розгортанні хмарних ЦОД.

2.1. Методологія управління ІТ-інфраструктурою центру оброблення даних провайдера хмарних послуг

Функціонування хмарних ЦОД характеризується множиною обставин, які обумовлені галузевими, географічними, міжнародними і політичними особливостями реалізації інформаційних процесів і надання послуг із сфери інформаційного обслуговування. Ці обставини впливають на склад і якість інформаційних і хмарних послуг, а також суттєво впливають на функціональність системи управління ресурсами ІТ-інфраструктури [349].

Ще однією важливою обставиною є суттєві характеристики хмарних обчислень [60], які впливають на формування вимог до функціональності системи управління хмарною інфраструктурою:

- *надання інструментів самообслуговування за вимогою споживача*, що означає розроблення і впровадження інтерфейсів доступу споживача хмарних послуг до інструментів управління споживанням послуг;
- *реалізація інструментів вимірювання споживання хмарних послуг,* що забезпечується розробленням і впровадженням систем моніторингу і тарифікації з боку провайдера хмарних послуг з метою інформування СХП про спожиті хмарні послуги;
- *надання мережевого доступу до хмарних послуг,* що означає організацію доступу споживачів хмарних послуг (СХП) за допомогою мережевих технологій і для широкого спектру пристроїв;
- *організація обчислювальних ресурсів у вигляді пулу і доступу до нього*, що означає надання можливості СХП отримувати інформаційні послуги від хмарної інфраструктури провайдера разом з іншими СХП без втручання в низькорівневі механізми і системи управління хмарною інфраструктурою;
- *швидка еластичність реалізації хмарних послуг,* означає можливість поступового збільшення або зменшення обсягу використовуваних ресурсів в автоматичному режимі реального часу залежно від потреб СХП і наявного навантаження.

Сформульовані в попередньому розділі задачі є надзвичайно складними і тому вимагають відповідного підходу під яким розуміємо методологію управління, в основі якої знаходиться сукупність принципів, підходів і інструментів підготовки, прийняття і реалізації рішень з управління IT-інфраструктурою хмарного ЦОД. За основу цього підходу вибираємо запропонований в [69, 351] декомпозиційно-компенсаційний підхід (ДКП), який добре себе зарекомендував для управління IT-інфраструктурами провайдерів ІКТ. Але розвиток галузі IT в останні роки виявив надзвичайно сильні тенденції до: поширення хмарних технологій, впровадження програмно-визначених систем, розгортання гіперконвергентних систем, застосування гібридних моделей прогнозування з використанням великих даних (Big Data), змінення моделей складних систем з плином часу. Таким чином, для розроблення методології управління та на її основі інформаційної технології з метою

вирішення поставлених задач беремо за основу ДКП, але з додатковими ключовими принципами:

- визначення стратегії управління ІТ-інфраструктурою в залежності від значень метрик і результатів прогнозування навантаження;
- впровадження операторної форми управління як необхідність автоматизації породження моделей і схем їх використання при управління ІТ-інфраструктурою;
- застосування схем реалізації стратегій;
- застосування прогнозування споживання ресурсів в умовах змінного навантаження і зміни стратегій управління;
- використання програмно-визначених і гіперконвергентних інфраструктур для обслуговування навантажень в хмарі.

В умовах хмарного ЦОД ДКП необхідно розвинути через врахування запропонованого в дисертації стратегічного рівня управління, ієрархічного управління і адаптації до поточного стану системи. Ще одним напрямком розвинення ДКП є поєднання управління ІТ-інфраструктурою по стану з управлінням традиційними методами математичного програмування з використанням прогнозування. Оскільки з'являється стратегічний рівень і зростає кількість моделей, які необхідно буде використовувати для планування і управління при кожній стратегії, то доцільним є формалізація управління використанням моделей на основі історії функціонування системи та результатів прогнозування. Реалізація методології управління може бути виконана за допомогою *операторного* підходу. Врахування вищеозначених аспектів дозволяє досягти заданої якості надання хмарних послуг в ієрархічній структурі ІТ-процесів з використанням достатньої кількості ресурсів рівня ІТ-інфраструктури та адаптуватися до змін навантаження через прогнозування споживання цих ресурсів.

Стратегія управління складається з обчислення метрик стану системи, прогнозування навантаження і планування схем реалізації стратегій. Для визначення стратегії управління на поточному кроці необхідно розробити моделі

і методи вибору стратегії в залежності від значення ключових метрик. Визначена стратегія управління впливає на параметри, критерії і обмеження моделей управління, які кожного разу перебудовуються, та на вибір схем реалізації стратегій управління з метою адаптації до поточних умов функціонування.

Схема ІТ-інфраструктури хмарного ЦОД, для якого розробляється інформаційна технологія управління (ІТУ), показана на рис. 2.1. Хмарний ЦОД являє собою ієрархічну програмно-апаратну систему, що складається з трьох рівнів, які відповідають трьом сервісним моделям надання хмарних послуг [349]. При цьому нижчий рівень – рівень інфраструктури (*рівень ресурсів*), переважно складається з апаратного забезпечення, а рівень платформи і рівень застосунків складаються з програмного забезпечення. Декомпозиція хмарного ЦОД як об’єкту управління виконана у розділі 3. Визначення ресурсів ІТ-інфраструктури наведено у п. 1.2.

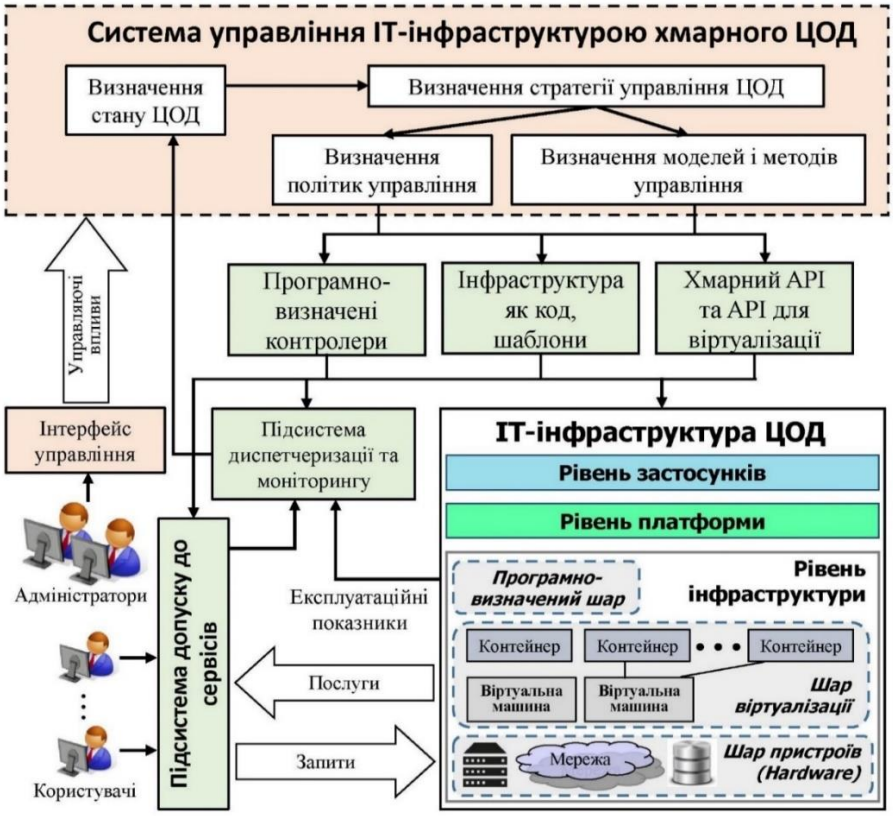


Рис. 2.1. Узагальнена схема управління ІТ-інфраструктурою хмарного ЦОД

СУІ реалізована за архітектурою мікросервісів і включає в себе: підсистему диспетчеризації, моніторингу і прогнозування; підсистему визначення стратегій управління і вибору методів управління; підсистему генерації керуючих впливів,

що впливають на програмно-апаратні засоби ЦОД. Експлуатаційні показники включають в себе дані моніторингу (фізичних і віртуальних) серверів, сховищ, мережевих пристроїв, системного і прикладного ПЗ на основі яких обчислюються метрики; якісні показники роботи сервісів; показники споживання електроенергії. Згенеровані керуючі впливи подаються через програмні інтерфейси (API) на виконавчі механізми, такі як пристрої увімкнення/вимкнення, гіпервізори, програмно-визначені контролери, балансувальники навантаження, контролери відмовостійкості і надійності. Задачею підсистеми визначення стратегій і методів управління є вироблення, на основі одержуваної інформації і значень метрик, рішень і сигналів керування, пов’язаних з оптимізацією функціональних режимів об’єктів ІТ-інфраструктури і розподілом ресурсів ІТ-інфраструктури між цими об’єктами в умовах змінних навантажень.

Одним з головних завдань при управлінні навантаженням і ресурсами ІТ-інфраструктури хмарного ЦОД є процес розміщення і перерозміщення (міграції) ВМ, який є функцією багатьох чинників, змінних, обмежень і критеріїв і являє собою комплекс оптимізаційних задач управління навантаженням в реальному часі для яких неможливо зафіксувати і прийняти рішення з управління на базі фіксованої групи моделей і методів [349].

Задача управління навантаженням і ресурсами ІТ-інфраструктури хмарного ЦОД характеризується такими факторами: *велика розмірність,* тобто кількість ВМ складає сотні тисяч, кількість ФС складає тисячі; *зміна структури і розмірності матриці* розподілу ВМ на ФС з часом; *динаміка навантажень і неможливість визначення їх розподілу*, тобто використання ресурсів змінюється на кожному кроці під впливом запитів клієнтів; *нелінійна залежність показників якості SLA від використаних ресурсів*. Таким чином, поставлена в дисертації задача належить до класу задач управління нелінійними нестаціонарними дискретними об'єктами зі змінною структурою (ННДОЗС).

Суттєво зменшити невизначеність при управлінні ресурсами ІТ-інфраструктури хмарного ЦОД дозволяє застосування прогнозування. Обґрунтування необхідності прогнозування навантаження на ресурси ІТ-

інфраструктури пов’язане з пошуком компромісу між бажанням вивільнити максимально можливу кількість ФС або залишити їх в активному стані з метою зменшення штрафів за порушення SLA, які виникають в результаті нестачі ресурсів. Точний прогноз дозволяє зменшити енергоспоживання і штрафи за рахунок прийняття випереджаючих управлінських рішень щодо розміщення нових ВМ та міграції існуючих [349]. Розробка і дослідження методів і моделей прогнозування споживання ресурсів в хмарному ЦОД представлені у розділі 4.

З урахуванням особливостей ІТ-інфраструктури сучасного хмарного ЦОД, завдань управління навантаженням і ресурсами, а також зазначених факторів розроблена узагальнена схема реалізації функцій управління (рис. 2.2), яка включає в себе: систему управління, що реалізує запропоновану інформаційну технологію; систему управління хмарою провайдера, яка базується на спеціалізованому ПЗ і забезпечує надання послуг користувачеві; мережу ЦОД; сховище ЦОД; множину ФС, які безпосередньо обробляють навантаження згідно сервісних моделей IaaS, PaaS, SaaS.

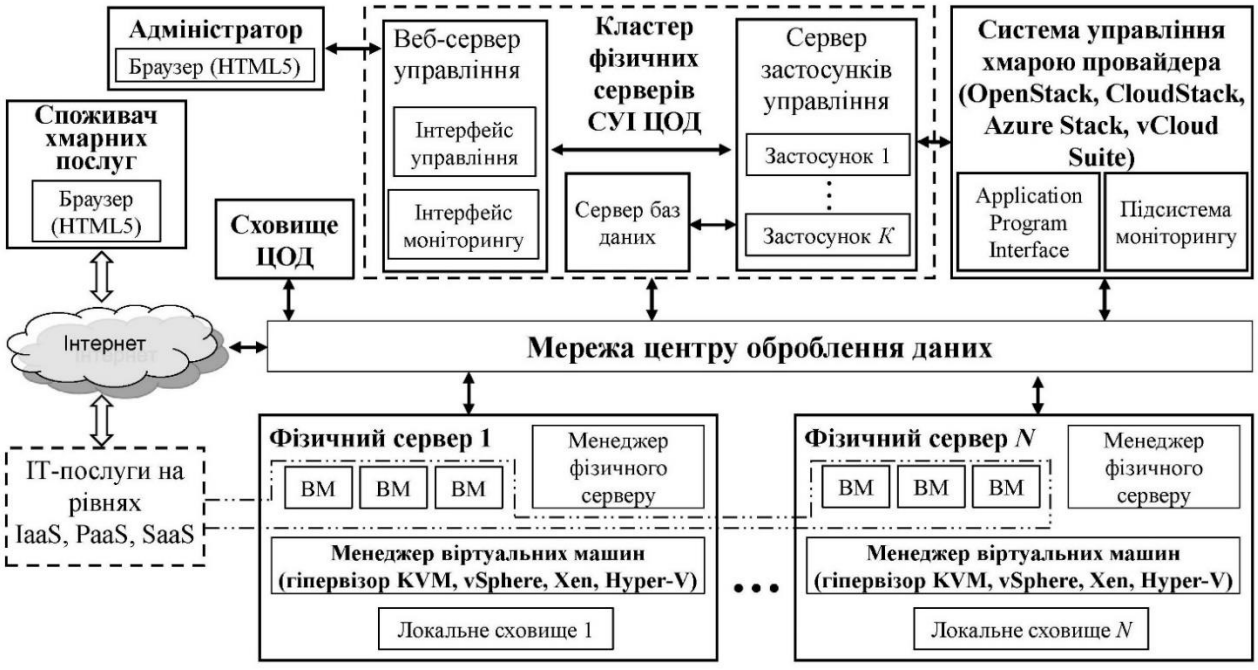


Рис. 2.2. Узагальнена схема реалізації функцій управління ІТ-інфраструктурою хмарного ЦОД

При цьому, система управління хмарними ресурсами провайдера реалізується на платформі з відкритим кодом, наприклад OpenStack, або на власній платформі провайдера, наприклад Azure Stack.

Отже, стан ІТ-інфраструктури хмарного ЦОД, як складної багаторівневої системи управління класу ННДОЗС, пропонується визначати в трьох вимірах: *вимір ресурсів* (надлишок або нестача); *вимір навантажень* (змінне або стале); вимір *динаміки навантажень* (з трендом на збільшення або на зменшення). Належність поточного стану до певних значень вимірів визначається сукупністю метрик на основі даних моніторингу і впливає на визначення стратегії управління ресурсами ІТ-інфраструктури.

Розглянемо принципові положення методології управління на основі означеного операторного підходу, який прийнятий для розроблення ІТУ ІТ-інфраструктурою хмарного ЦОД. При цьому, в залежності від поточного стану хмарного ЦОД, визначеної стратегії управління і результатів прогнозування навантаження склад критеріїв і обмежень змінюються під впливом технологічних особливостей хмарного ЦОД і необхідності надання сервісів із зазначеними показниками якості. Таким чином, необхідно розробити операторну форму управління ресурсами ЦОД, схеми і стратегії її реалізації, а також моделі визначення стратегій залежно від історичних даних, значень метрик та результатів прогнозування.

2.2. Операторна форма постановки, аналізу і розв'язання задач управління ІТ-інфраструктурою

Сформулюємо загальну постановку задачі управління ресурсами ІТ-інфраструктури хмарного ЦОД, яка відноситься до класу задач управління ННДОЗС.

Для кожного кроку управління t визначимо матрицю $\mathbf{A}: m \times n$ (2.1), елементами якої є цілочисельні змінні $a_{ij}(t) \in \{0,1\}$, які вказують, чи працює j-а ВМ з множини V на i-му ФС з множини P ($a_{ij}(t) = 1$), або не використовується ($a_{ij}(t) = 0$).

$$\mathbf{A}(t)=\begin{bmatrix} a_{11}(t) & a_{12}(t) & \cdots & a_{1n}(t) \\ a_{21}(t) & a_{22(t)} & \cdots & a_{2n}(t) \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1}(t) & a_{m2}(t) & \cdots & a_{mn(t)} \end{bmatrix} \quad (2.1)$$

Матриця **A** позначає загальний стан системи (ЦОД). Кількість змін значень в стовпчиках матриці **A** з 0 на 1 дорівнює кількості завершених міграцій ВМ між ФС. Тобто, для кожного n виконується рівняння $\sum_{i=1}^{m} a_{ij} = 1$. У загальному випадку $n=N(t)$ і $m=M(t)$, тобто кількість ВМ і ФС може змінюватись. Визначимо змінні, що позначають вхід СУІ: $q(t)$ – кількість нових ВМ для розміщення; $r(t)$ – кількість ВМ, які зупинені. Визначимо змінні, що позначають управляючі впливи СУІ: $p(t)$ – кількість ВМ, які мігрують; $k(t)$ – кількість увімкнених/вимкнених ФС. Визначимо збурюючі впливи $\delta(t+1), \varepsilon(t+1)$ – кількість ФС і ВМ, які вийшли з ладу, $\delta, \varepsilon \geq 0$. Тоді наступний стан системи визначається рівнянням $\mathbf{A}(t+1) = F\left[\mathbf{A}(t), \hat{N}(t+1), U(t), Z(t)\right]$, де $F[\bullet]$ є функціоналом управління, який реалізується обраною стратегією управління; $U(t) = \{k(t), p(t)\}$ – керуючі впливи; $\hat{N}(t+1) = N(t) + \hat{q}(t) - \hat{r}(t) - \varepsilon(t+1)$ – кількість ВМ на наступному кроці управління, $Z(t)$ – план міграцій ВМ між ФС. Кількість ФС на наступному кроці визначається рівнянням $M(t+1) = M(t) + k(t) - \delta(t+1)$. Критерії, які визначаються обраною стратегією управління: мінімізація кількості активних ФС, $M(t)$; мінімізація міграцій ВМ, $p(t)$; мінімізація кількості випадків повного завантаження ресурсу; мінімізація затримки розгортання ВМ та ін. Обмеження, які необхідно враховувати при реалізації різних стратегій: кількість вхідних/вихідних міграцій на ФС, обмеження використання кожного ресурсу в межах ФС, обмеження швидкості мережевої взаємодії, обмеження швидкості роботи з даними в СЗД та ін.

При цьому, в залежності від поточного стану хмарного ЦОД і визначеної стратегії управління, склад критеріїв і обмежень змінюються під впливом технологічних особливостей хмарного ЦОД та необхідності надання сервісів із зазначеними показниками якості. Таким чином, необхідно кожного разу будувати нові моделі хмарного ЦОД і застосовувати різні алгоритми і методи

управління у вигляді схем реалізації з метою адаптації до поточних умов функціонування.

Необхідність розроблення операторної форми постає під впливом низки чинників, притаманних розв'язанню проблем управління ІТ-інфраструктурою. Необхідно зауважити, що ця ідея виникла під впливом досвіду ефективного застосування потужних концепцій об'єктно-орієнтованого програмування і теорії автоматичного управління. Дійсно, концепції ООП, такі як клас і об'єкт, характеристики поведінки об'єктів за належністю до класу (функції або методи), ієрархія класів, поліморфізм і перевантаження методів, успадкування характеристик і поведінки спонукають до структурування програм, побудови системи компонентів, які мають стати будівельними блоками на кожному рівні управління ІТ-інфраструктурою.

Введемо чинники, які впливають на операторну форму. По-перше, це наявність декількох рівнів управління ІТ-інфраструктурою, обумовлена її складністю і багатофункціональністю. Виглядає доцільним попередити управління вибором стратегії. А оскільки стратегій декілька і результати їх можна оцінити тільки після планування, то операторна форма є дуже зручною. По-друге, це багатокритеріальність, коли власне критерій є предметом вибору.

Враховуючи вищезазначене, цільова функція управління ІТ-інфраструктурою ЦОД як об'єктом класу ННДОЗС в операторній формі представлена у такий спосіб [349]:

$$\min\left[\mathbf{C}_1\mathbf{F}_i\left[P,V\right]+\mathbf{C}_2\mathbf{M}_i\left[P,V\right]+\mathbf{C}_3\mathbf{L}_i\left[P,V\right]\right] \qquad (2.2)$$

де $\mathbf{F}_i\left[P,V\right]=(b_1^F,b_2^F,...,b_m^F)^T$ – оператор *i*-ї стратегії, що визначає ФС з множини *P* з когерентними ВМ з множини *V*, $b^F\in\{0,1\}$;

$\mathbf{M}_i\left[P,V\right]=(b_1^M,b_2^M,...,b_m^M)^T$ – оператор *i*-ї стратегії, що визначає розподіл некогерентних ВМ з множини *V* між ФС з множини *P*, $b^M\in\{0,1\}$;

$\mathbf{L}_i\left[P,V\right]=(b_1^L,b_2^L,...,b_m^L)^T$ – оператор *i*-ї стратегії, що визначає ФС з множини *P* які переведені в режим сну, $b^C\in\{0,1\}$;

$\mathbf{C}_1=(c_1^1,c_2^1,...,c_m^1)$ – вектор коефіцієнтів витрат на підтримку роботи ФС;

$\mathbf{C}_2=(c_1^2,c_2^2,...,c_m^2)$ – вектор коефіцієнтів витрат на підтримку роботи ФС, виділених для обслуговування некогерентних ВМ;

$\mathbf{C}_3 = (c_1^3, c_2^3, ..., c_m^3)$ – вектор коефіцієнтів витрат на підтримку ФС у вимкнутому стані.

Результатом дії операторів **F** і **M**, які отримують на вході вектори характеристик ФС і ВМ, є матриця розподілу ВМ між ФС **A**(*t*+1). Результатом дії оператора **L**, який отримує на вході метрики стану хмарного ЦОД, є план міграцій ВМ *Z*(*t*) і керуючі впливи *U*(*t*). Результатом цільової функції управління є значення операційних витрат на підтримку роботи хмарного ЦОД.

Результати проведених експериментів і реалізація системного підходу до проектування системи керування інформаційними процесами на якісному рівні дозволяють сформулювати таке твердження.

Твердження 1. Про функціональну повноту системи керування.

Створена інформаційна технологія управління серверами на основі множини запропонованих методів у складі схем реалізації керуючої системи забезпечує достатній рівень якості попередньої обробки експериментальних даних, адаптивного оцінювання структури і параметрів математичних моделей з урахуванням наявних невизначеностей структурного, параметричного і стохастичного типів, обчислення оцінок прогнозів та керуючих впливів гарантують отримання заданої якості керування множиною серверів *J*(*U*(*t*)).

Обґрунтування. Нехай якість керування задана монотонною функцією $Q = f(q_1, q_2, ..., q_m)$, де q_i – якість роботи *i*-го функціонального сервера (ФС) на рівні інфраструктури інформаційної системи. Монотонність функції $Q = f(q_1, q_2, ..., q_m)$ випливає з результатів експериментальних досліджень, якщо $q_l' > q_l$, то $Q' \geq Q$, де q_l' – значення якості роботи *l*-го сервера на наступному кроці управління, Q' – значення якості керування на наступному кроці управління. Доведемо, що розроблена ІТ дозволяє забезпечити якість роботи одного, будь-якого ФС, а значить і всієї множини ФС хмарного центру оброблення даних (ЦОД).

ФС *i* може перебувати в одному з трьох станів: s_i^1 – в режимі сну, s_i^2 – в режимі обслуговування когерентних ВМ з множини *V* і s_i^3 – в режимі обслуговування некогерентних ВМ з множини *V*. Якість необхідно забезпечити

для станів s_i^2 і s_i^3. Для стану s_i^2 за рахунок виконання відповідних методів в складі схем реалізації допустимих стратегій управління, ВМ вже підібрані так, що не відбувається перевантаження ресурсів *i*-го ФС, і, відповідно, забезпечується задана якість обслуговування усіх сервісів і ВМ на рівнях сервісних моделей IaaS, PaaS і SaaS. Таким чином, розглянемо забезпечення якості управління тільки для стану s_i^3.

Нехай схема реалізації *j*-ї стратегій управління S_j в складі розробленої ІТ забезпечуються функціонуванням методів управління $W = \{W_1, W_2, ..., W_k\}$, $0 < k < K$, де K – максимальна кількість забезпечуючих методів управління в складі ІТ. Експериментально визначена *j*-а комбінація методів з множини W є схемою реалізації визначеної стратегії управління S_j в *j*-му стані хмарного ЦОД. Експериментальне дослідження методів W та логіки функціонування системи довели, що для допустимих станів кожного *i*-го ФС з множини P застосування будь-якого методу з множини W забезпечує за скінченний час перехід *i*-го ФС у стан, який не є перевантаженим, а значить, забезпечується задана якість управління *i*-м ФС q_i.

Введемо поняття схеми реалізації оператора, яка залежить від історичних даних, що збираються в системи в процесі управління, від значень метрик та від результатів прогнозування. При цьому, операторна схема дає механізми, які потім реалізуються відповідно до стратегій. Моделі управління, склад обмежень і критеріїв визначаються в процесі управління в залежності від історичних даних і результатів прогнозу.

2.3. Комплекс схем реалізації операторної форми управління ІТ-інфраструктурою у просторі визначених стратегії

Операторний підхід до управління ІТ-інфраструктурою хмарного ЦОД в цілому реалізується послідовно на трьох рівнях: стратегічному рівні, рівні планування і рівні управління. Рівень стратегій визначає такі стратегії управління: з нестачею ресурсів при сталому навантаженні, з надлишком ресурсів при сталому навантаженні, з нестачею ресурсів і трендом на зменшення

навантаження, з надлишком ресурсів і трендом на зменшення навантаження, з нестачею ресурсів і трендом на збільшення навантаження, з надлишком ресурсів і трендом на збільшення навантаження. Визначення стратегії управління відбувається через реалізацію трьох стадій: оцінки стану IT-інфраструктури за допомогою метрик, отримання завдань і запитів з боку користувача та прогнозування навантаження [349].

Кожна стратегія управління реалізується вирішенням таких завдань планування: обчислення метрик, які відображають тенденції навантаження і складу IT-послуг, що надаються провайдером в поточний час; прогнозування навантаження на трьох рівнях сервісної моделі хмарних обчислень; визначення оптимізаційних методів, які необхідно застосувати в поточний час, за допомогою операторного підходу з урахуванням специфіки ЦОД провайдера, складу методів оптимізації і поточних умов роботи ЦОД [349]. Результатом роботи рівня планування є підмножина завдань управління, реалізація яких призводить до досягнення мети визначеної стратегії управління. Визначені завдання реалізуються за допомогою програмно-визначених методик і технологій на трьох рівнях сервісної моделі хмарних обчислень [349].

Підмножина завдань управління для визначеної стратегії реалізується схемами реалізації і методами, розробленими в розділах 3, 4, 5. Застосування того чи іншого методу управління ресурсами або їх комбінацій в залежності від стану ресурсів і навантажень ЦОД, а також визначеної стратегії показано в табл. 2.1.

Табл. 2.1. Вибір стратегії і схеми реалізації управління ресурсами у залежності від стану ЦОД

Передумови	**Стратегії**	**Схеми реалізації стратегій**	**Базові методи**
Середній коефіцієнт життєздатності віртуальної машини; індикатор дисбалансу фізичного сервера:	Управління з нестачею ресурсів при сталому навантаженні, S_1	Рівномірна консолідація ВМ.	Модифікований метод відпалу.
	Управління з надлишком ресурсів при сталому навантаженні, S_2	Управління міграцією та розміщенням ВМ в сталому режимі. Двостадійний метод управління ресурсами.	Модифікований метод променевого пошуку.

Передумови	Стратегії	Схеми реалізації стратегій	Базові методи
	Управління з нестачею ресурсів і трендом на зменшення навантаження, S_3	Інтегроване управління ресурсами (ІУР).	Метод інтегрованого управління ресурсами.
	Управління з надлишком ресурсів і трендом на зменшення навантаження, S_4	Інтегроване управління ресурсами.	Метод інтегрованого управління ресурсами. Метод управління потужністю ЦОД.
	Управління з нестачею ресурсів і трендом на збільшення навантаження, S_5	Динамічна консолідація і розміщення ВМ.	Модифікований метод навчання з підкріпленням.
	Управління з надлишком ресурсів і трендом на збільшення навантаження, S_6	Динамічна консолідація і розміщення ВМ з управлінням потужністю.	Модифікований метод навчання з підкріпленням. Метод управління потужністю ЦОД.

Схеми реалізації оператора залежать від історичних даних, від значень метрик та від результатів прогнозування і являють собою певну визначену стратегію управління. Параметри моделі управління, склад обмежень і критеріїв визначаються в процесі управління. Стратегії S управління визначаються множиною моделей $\Omega(D,O,G)$ і методів управління з множиною змінних D, множиною обмежень O і множиною критеріїв G які визначаються в процесі функціонування системи. Кожна i-та стратегія управління $S_i = \Omega(D_i, O_i, G_i), D_i \in D, O_i \in O, G_i \in G$ реалізує певний оператор цільової функції (2.2). При цьому кожна підмножина складається з довільної кількості елементів, $D_i = \{d_{0,i}, d_{1,i}, \ldots\}$, $O_i = \{o_{0,i}, o_{1,i}, \ldots\}, G_i = \{g_{0,i}, g_{1,i}, \ldots\}$, $D_i \neq \varnothing, O_i \neq \varnothing, G_i \neq \varnothing \ \ \forall i$.

Таким чином, схема реалізації операторів цільової функції (2.2) визначається системою.

$$\begin{cases} \mathbf{F}_i[P,V]: \langle \ \sum_{i=1}^{N(t)} res_i^{VM}, \sum_{j=1}^{M(t)} Res_j^{PM}, \Omega(D_i, O_i, G_i) \ \rangle \to \mathbf{A}(t+1) \\ \mathbf{M}_i[P,V]: \langle \ K, \Omega(D_i, O_i, G_i) \ \rangle \to M(t+1) \\ \mathbf{L}_i[P,V]: \langle \ K, \Omega(D_i, O_i, G_i) \ \rangle \to Z(t) \end{cases}, \qquad (2.3)$$

де $\mathbf{F}_i$, $\mathbf{M}_i$, $\mathbf{L}_i$ – оператори цільової функції (2.2), res_i^{VM} – ресурси, які споживає i-та ВМ, Res_i^{PM} – ресурси, які споживає j-й ФС, K – множина метрик оцінювання стану хмарного ЦОД, $\Omega(D_i, O_i, G_i), D_i \in D, O_i \in O, G_i \in G$ – вибрана модель в процесі управління ресурсами хмарного ЦОД. Конкретні значення D_i, O_i, G_i вибираються відповідно до конфігурації хмарного ЦОД, значень табл. 2.1 і вимог реалізації кожного методу управління.

Розглянемо методи і моделі автоматичного визначення стратегії управління в залежності від метрик оцінки стану хмарного ЦОД на базі моделей нечіткого виводу і моделі "гри з природою".

2.4. Моделі визначення стратегій управління

З метою автоматичного визначення стратегії управління необхідно розробити модель і метод нечіткого виводу, а також модель "гри з природою", які дозволять визначити стратегію управління в залежності від метрик оцінки стану хмарного ЦОД.

Розглянемо модель нечіткого виводу, в якій стратегія управління є вихідною змінною X, а метрики оцінки стану хмарного ЦОД представлені n вхідними змінними множини метрик K. В якості вхідних змінних взяті: K_1 – середній коефіцієнт життєздатності ВМ, K_2 – індикатор дисбалансу ФС, K_3 – коефіцієнт відношення необхідних ресурсів до середнього об’єму наявних ресурсів, K_4 – поріг вільних ресурсів та K_5 – метрика ємності ЦОД. При цьому, метрики K_1, K_2 K_3 вперше запропоновані в дисертації. Вихідна змінна приймає значення, відповідні назвам стратегій: S_1 – з нестачею ресурсів при сталому навантаженні, S_2 – з надлишком ресурсів при сталому навантаженні, S_3 – з нестачею ресурсів і трендом на зменшення навантаження, S_4 – з надлишком ресурсів і трендом на зменшення навантаження, S_5 – з нестачею ресурсів і трендом на збільшення навантаження, S_6 – з надлишком ресурсів і трендом на збільшення навантаження. За допомогою правил виду "Якщо $K_1 \in A_1 \wedge K_2 \in A_2 \wedge \ldots \wedge K_n \in A_n$, то $X \in S_i$ ", де A_n – множина значень вхідної змінної K_n, реалізований нечіткий вивід Мамдані для визначення стратегії управління ЦОД на поточному кроці управління. Для побудови функцій належності використовується підхід,

представлений в роботі [321]. Зазначений підхід модифікований через розроблення нової методики вибору піків, яка полягає у поєднанні декількох сусідніх інтервалів для визначення піку. При цьому використовуються історичні дані роботи системи.

Розглянемо модель "гри з природою", де перший гравець – модуль вибору стратегії управління (гравець МВСУ) хмарним ЦОД, а другий гравець – "природа" у вигляді навантаження на ресурси хмарного ЦОД (гравець ЦОД). При цьому "природа" не є союзником або супротивником першого гравця. Першим етапом розроблення цієї моделі є побудова матриці виграшів. Гравець МВСУ має шість стратегій S_i, $i=\overline{1,6}$ (табл. 2.1), гравець ЦОД може знаходитись в одному із n станів A_j, $i=\overline{1,n}$. Стани гравця ЦОД визначаються комбінаціями значень таких метрик: K_1 – середній коефіцієнт життєздатності ВМ, який характеризує рівень навантаження; K_2 – коефіцієнт прямої тренду навантаження, обчислений з використанням історичних даних миттєвого коефіцієнту життєздатності ВМ, якщо; K_3 – метрика ємності ЦОД, яка характеризує надлишок або нестачу ресурсів. Значення змінних стану можуть бути такими: при $0.95 \le K_1 \le 1.05$ навантаження стале, інакше змінне; якщо $K_2 \le -0.1$, то навантаження зменшується, якщо $K_2 \ge 0.1$, то навантаження збільшується, інакше навантаження вважається сталим, $-0.1 < K_2 < 0.1$; якщо $K_3 \le \delta$, де δ – це поріг використання ресурсів, який визначається експертним шляхом, то є надлишок ресурсів, інакше існує нестача ресурсів в ЦОД. Виграші від застосування стратегії S_i в стані A_j позначимо q_{ij}. Кодування стратегій в залежності від значень метрик K показано в табл. 2.2.

Табл. 2.2. Кодування станів гравця ЦОД в "грі з природою"

Наявність ресурсів	Навантаження	Тренд зміни навантаження	Стан гравця ЦОД
$K_3 > \delta$	$0.95 \le K_1 \le 1.05$	$-0.1 < K_2 < 0.1$	A_1
$K_3 \le \delta$	$0.95 \le K_1 \le 1.05$	$-0.1 < K_2 < 0.1$	A_2
$K_3 > \delta$	$0 < K_1 < 0.95, K_1 > 1.05$	$K_2 \le -0.1$	A_3
$K_3 \le \delta$	$0 < K_1 < 0.95, K_1 > 1.05$	$K_2 \le -0.1$	A_4
$K_3 > \delta$	$0 < K_1 < 0.95, K_1 > 1.05$	$K_2 \ge 0.1$	A_5
$K_3 \le \delta$	$0 < K_1 < 0.95, K_1 > 1.05$	$K_2 \ge 0.1$	A_6

Враховуючи вищезазначене, матриця виграшів Q приймає такий вигляд (табл. 2.3). Початкові значення виграшів q_{ij} обираються через експертні оцінки методом Дельфі.

Табл. 2.3. Матриця виграшів "гри з природою"

Схема реалізації стратегії управління ресурсами ЦОД	Стратегія гравця МВСУ	Стан гравця ЦОД					
		A_1	A_2	A_3	A_4	A_5	A_6
Метод рівномірної консолідації ВМ	S_1	q_{11}	q_{12}	q_{13}	q_{14}	q_{15}	q_{16}
Метод управління міграцією та розміщенням ВМ в сталому режимі. Двостадійний метод управління ресурсами	S_2	q_{21}	q_{22}	q_{23}	q_{24}	q_{25}	q_{26}
Метод інтегрованого управління ресурсами	S_3	q_{31}	q_{32}	q_{33}	q_{34}	q_{35}	q_{36}
Метод інтегрованого управління ресурсами	S_4	q_{41}	q_{42}	q_{43}	q_{44}	q_{45}	q_{46}
Метод динамічної консолідації і розміщення ВМ. Метод управління потужністю ЦОД	S_5	q_{51}	q_{52}	q_{53}	q_{54}	q_{55}	q_{56}
Метод динамічної консолідації і розміщення ВМ. Метод управління потужністю ЦОД	S_6	q_{61}	q_{62}	q_{63}	q_{64}	q_{65}	q_{66}

Виграші q_{ij} гравця ЦОД в процесі управління визначаються на основі історичних даних за результатами обчислення цільової функції (2.2). В якості критерія вибору оптимальних рішень вибраний максимінний критерій Вальда. При застосуванні цього критерію вибирається стратегія, яка гарантує виграш не менший, ніж "нижня ціна гри". Тобто, з усіх найневдаліших результатів вибирається найкращий (позиція крайнього песимізму).

2.5. Структурно-функціональна модель системи управління ІТ-інфраструктурою хмарного центру оброблення даних

2.5.1. Підстави інтегрованого управління ресурсами ІТ-інфраструктури

Парадигма хмарних обчислень кардинально змінює не тільки погляд на надання ІТ-послуг, але і на розуміння самих ІТ-послуг. Як ІТ-послуги, на сьогодні розглядаються ПЗ, платформи, ІТ-ресурси тощо. Це, в свою чергу,

зажадало змін і принципів організації інфраструктури надання ІТ-послуг. У сучасних ЦОД, що є базою для надання хмарних обчислень, використовуються технології віртуалізації, програмно-визначені мережі, інноваційні підходи в області оброблення і зберігання інформації, а також інші рішення, що дозволяють надавати різноманітні ІТ-послуги користувачам і замовникам. При цьому істотно зростає роль системи управління ІТ-інфраструктурою ЦОД, яка повинна не тільки підтримувати якість ІТ-послуг на узгодженому рівні при змінній динаміці запитів численних користувачів, постійно зростаючої складності інформаційних процесів, безупинному збільшенні обсягів оброблюваної інформації, але і ефективно використовувати дорогі обчислювальні і комунікаційні ресурси ЦОД через зменшення операційних витрат.

Сучасна ІТ-інфраструктура ЦОД характеризується великою кількістю ресурсів, широким використанням віртуалізації, збільшенням складності, істотною динамікою технологічних змін, збільшенням обсягу оброблюваної інформації.

При розв’язанні задач забезпечення ефективного функціонування ІТ-інфраструктури ЦОД провайдера хмарних послуг СУІ повинна враховувати такі особливості ІТ-інфраструктури хмарного ЦОД:

- великі об’єми ресурсів кожного типу або виду;
- на серверах розгортаються ВМ і контейнери, на яких встановлюються застосунки, що підтримують сервіси;
- ІТ-інфраструктура є більш статичною і менш схильна до змін, ніж застосунки і програмні засоби;
- зміни в ІТ-інфраструктурі ініціюються необхідністю змін на рівні застосунків і бізнес-процесів.
- кожен сервер використовується одночасно декількома застосунками;
- якщо поверх ІТ-інфраструктури ЦОД функціонують застосунки замовників, то можливий конфлікт інтересів адміністраторів серверів і адміністраторів застосунків.

Дослідження [83, 134] показують, що сучасні підходи і методи управління ЦОД повинні враховувати використання програмно-визначених технологій, непередбачувану інтенсивність навантажень і зростання споживання ІТ-ресурсів при появі нових сервісів. Останнім часом широко впроваджується програмно-визначений підхід для вирішення задач контролю та управління, які існують у традиційних дата-центрах.

Операційні витрати на управління ресурсами в сучасних ЦОД є великими і постійно зростають [239], а застосування парадигми програмно-визначених пристроїв на всіх рівнях управління дозволяє зменшити витрати на адміністрування, розширити можливості інтеграції і забезпечити масштабування СУІ.

У хмарних ЦОД вирішуються такі задачі: надання ресурсів, розподіл ресурсів, відображення ресурсів, масштабування ресурсів, оцінка ресурсів і посередництво в ресурсах в умовах невизначеності і непередбачуваності навантажень. При створенні СУІ необхідно розробляти моделі і вирішувати завдання надання, розподілу, масштабування і оцінки ІТ-ресурсів. Це необхідно робити з урахуванням того, що провайдери хмарних послуг, з одного боку, несуть відповідальність за відповідність поточних значень показників якості послуг, що надаються узгодженим з замовником значенням (наприклад, часу відгуку, затримки, ймовірності втрати запитів і ін.). З іншого боку, провайдери зацікавлені в наданні високоякісних послуг з мінімальним використанням пропускної спроможності каналів зв'язку, дискового простору для зберігання даних і інших задіяних для надання послуг ІТ-ресурсів також з мінімальною витратою енергоресурсів.

У той же час, постачальники послуг зобов'язані надавати ІТ-послуги з показниками якості, зафіксованими у відповідній угоді про рівень надання послуг SLA. Для виконання вимог SLA провайдери послуг часто змушені резервувати ресурси, запускати додаткові екземпляри додатків і проводити інші заходи, що передбачають надмірне використання ІТ-ресурсів. Оплату за таку надмірність провайдери перекладають на своїх клієнтів, що в свою чергу призводить до підвищення вартості послуг і робить таких провайдерів

неконкурентоспроможними. У даному випадку у виграші виявляються ті провайдери, які задіють у СУІ механізми, які відстежують тенденції збільшення запитів користувачів і динамічно збільшують або зменшують надання ресурсів за результатами прогнозування. Таким чином, планування ресурсів і політика їх розподілу безпосередньо впливають на вартість хмарних послуг.

Зниження якості ІТ-послуги, що сталося, наприклад, унаслідок збільшення кількості запитів користувачів, може бути компенсовано збільшенням обсягів ресурсів, виділених для реалізації цієї послуги. Тому при створенні СУІ особлива увага приділяється вирішенню завдань управління ресурсами ЦОД з урахуванням частих змін навантаження на ВМ і різних умов функціонування ВМ. Причому завдання розміщення ВМ і їх міграції повинні вирішуватися в режимі онлайн. Алгоритм перерозподілу навантаження на ФС, що керує міграцією ВМ в режимі онлайн, представлений в п. 5.6.

Політика планування і розподілу обчислювальних ресурсів безпосередньо впливає на вартість та продуктивність хмарних послуг. Вони відіграють важливу роль у наданні хмарних послуг у спосіб, що супроводжується досягненням заданої продуктивності, відповідністю вимогам SLA, ефективним використанням ресурсів, енергозбереженням та збільшенням прибутку хмарних провайдерів. Важливим етапом у процесі управління є моніторинг користувацьких послуг та ІТ-інфраструктури ЦОД з метою прогнозування і завчасного виділення достатніх ресурсів для роботи служб відповідно до умов SLA. Застосування засобів моніторингу обґрунтовано в п. 7.6.

Для реагування на виникаючі інциденти та збурюючі впливи СУІ повинна виробляти відповідні рішення управління. Враховуючи складність ІТ-інфраструктури, кількість послуг та функцій, а також їх взаємозв’язок, керуючі впливи та рішення на різних рівнях управління часто є несвоєчасними та неефективними. Часто, оператори вручну здійснюють такі коригувальні дії: зміни в конфігурації мережі, зміни в конфігурації пам'яті, міграцію ВМ, зміну конфігурації ВМ та інші. Тому провайдерам хмарних послуг необхідно використовувати інтегровані системи управління, які автоматизують ці процеси.

Традиційні підходи до управління ресурсами ЦОД і, зокрема, до управління ВМ не враховують повною мірою динаміку процесів, що відбуваються, і зазвичай виробляють керуючі впливи, коли інциденти і проблеми вже виникли. Тому запропоновано інтегрований метод керування розміщенням і міграцією ВМ (розділи 5 і 7), який заснований на врахуванні динаміки процесів, що відбуваються в ЦОД, при розгортанні нових ВМ одночасно з міграцією ВМ, що вже працюють.

Масштабованість ресурсів ЦОД також набуває важливого значення, оскільки в епоху цифрових змін кількість клієнтів починає зростати в кожному бізнесі. Проблема масштабованості, що стосується планування потужності, може бути визначена як здатність IT-інфраструктури зростати пропорційно робочому навантаженню, не додаючи складності або труднощів при управлінні нею. Для вирішення проблеми масштабованості в дисертаційній роботі пропонується в розробленій інформаційній технології використовувати архітектурні моделі веб-масштабування (web-scale) і гіперконвергенції, розроблені у розділі 7.

Значний вплив на ефективність процесу управління ресурсами ЦОД надає використання історичних даних і передбачення потреби в ресурсах на короткострокову або середньострокову перспективу. У запропонованому в розділі 4 адаптивному методі комбінованого прогнозування навантаження на обчислювальні ресурси хмарного ЦОД використовується середньострокове передбачення навантаження на ресурси ФС і враховується інтенсивність споживання ресурсів кожною ВМ, що дозволяє підвищити якість прийнятих рішень, пов'язаних з міграцією ВМ, розміщенням нових ВМ і видаленням ВМ при зниженні навантаження.

Крім того, кілька поколінь апаратних засобів від різних постачальників співіснують в сучасних ЦОД, тому ємність ФС неоднорідна. Потреби ВМ в ресурсах також неоднорідні у зв'язку зі зміною набору застосувань, якими вони керують. Дослідження показують, що ФС в багатьох ЦОД часто виявляються недостатньо завантаженими внаслідок виділення надмірних резервів ресурсів на покриття пікового навантаження або через великі інтервали часу між оптимізацією розміщення ВМ в автономному режимі [73]. Розроблена у розділі

3 модель хмарного ЦОД враховує гібридність і гетерогенність апаратно-програмного забезпечення.

Таким чином, актуальною є розроблення інформаційної технології управління IT-інфраструктурою ЦОД провайдера хмарних послуг і архітектури програмно-визначеної системи управління ресурсами на основі операторного підходу з урахуванням прогнозу навантаження.

2.5.2. Задачі управління IT-інфраструктурою

Провайдер хмарних послуг надає різні за змістом і призначенням сервіси для клієнтів. Клієнти провайдера можуть надавати деякі сервіси для інших клієнтів через реалізацію багатокористувацького багатоланкового ЦОД, який представлений складною мережною і обчислювальною інфраструктурою [271, 289]. Між провайдерами хмарних послуг укладаються угоди з надання послуг на базі декількох ЦОД. Для цього визначаються відповідні стратегії, схема реалізації яких дозволяє визначити управляючі впливи при наданні ресурсів на вимогу.

Для надання IT-сервісів провайдери послуг зазвичай використовують одну або кілька ВМ, які розміщуються на ФС. При цьому необхідно визначити стратегії управління або політики управління, залежно від значень метрик і результатів прогнозу. При погіршенні якості послуг внаслідок збільшення навантаження на ВМ використовуються методи горизонтального або вертикального масштабування. Для таких варіантів надання IT-послуг пропонується схема системи централізованого управління, наведена на рис. 2.3.

Об'єкт управління складається з множини ФС, кожен з яких характеризується відповідним апаратним забезпеченням з встановленою операційною платформою і має фіксовану ємність ресурсів, яка характеризується продуктивністю процесорів, об'ємом ОП і сховища, пропускною спроможністю мережевого підключення, продуктивністю роботи дискової підсистеми та ін. На кожному сервері розміщується одна або кілька ВМ. Сервери з'єднуються в групи і, в кінцевому підсумку, утворюють обчислювальну базу ЦОД. Комунікаційна мережа ЦОД забезпечує взаємодію серверів на рівнях

моделі OSI L2 та L3 і на цей час створюється з використанням програмно-визначених технологій і підтримкою OpenFlow. Сервера фізично групуються у вигляді стійок і підключаються до комутатора стійки (ToR Switch).

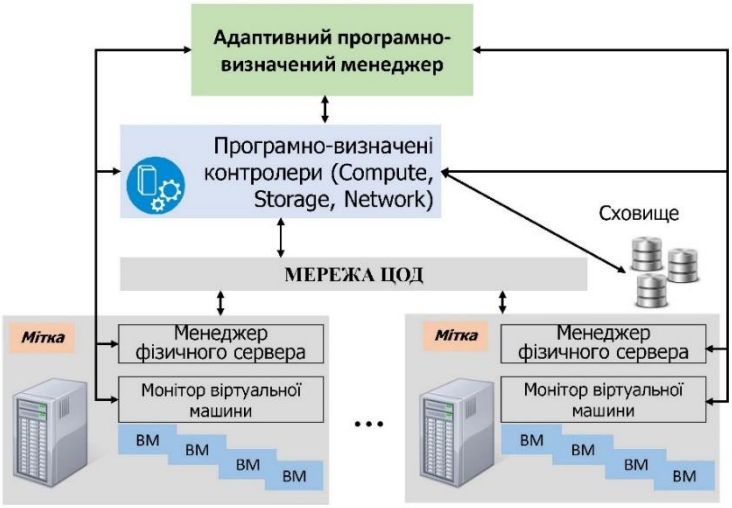


Рис. 2.3. Структурна схема системи централізованого управління ресурсами хмарного ЦОД

ЦОД з підтримкою хмарних послуг орієнтований на оброблення пакетного і трансакційного навантаження. Проблеми забезпечення заданої якості надання IT-послуг в гетерогенному IT-середовищі вирішуються через використання сервісних моделей IaaS, PaaS і SaaS [60]. Модель IaaS є основою для надання базових сервісів віртуалізації і функціонування моделей PaaS і SaaS. Зберігання даних забезпечується централізованим сховищем, а також однією з технологій розподіленого зберігання, таких як мережева файлова система (NFS), General Parallel File System (GPFS), Hadoop distributed file system від Yahoo! (HDFS) [252], WAS [52], GoogleFS [51], GlusteFS, Ceph FS і аналогічні.

Основним завданням при управлінні хмарним ЦОД є ефективний розподіл ресурсів між кінцевими користувачами послуг в умовах виникнення збоїв, непередбачуваної зміни навантаження, кількості клієнтів і їх запитів. Під ефективним розподілом ресурсів будемо розуміти такий розподіл ресурсів, при якому дотримується заданий рівень послуг SLA і витрачається мінімальна кількість енергії на забезпечення роботи серверів та систем зберігання даних. Задачі управління IT-інфраструктурою розбиті на рівні і вирішуються за допомогою двох стадій (табл. 2.4).

Табл. 2.4. Задачі і стадії управління IT-інфраструктурою хмарного ЦОД

Рівень управління IT-інфраструктурою	**Вирішувані задачі**
ЦОД, хмарний ЦОД, федеративні ЦОД (Peer-to-Peer Management)	Пошук ресурсу для виконання задач клієнтів за критеріями: виконання SLA, ціна.
Рівень ЦОД. Адаптивний програмно-визначений менеджер.	Визначення стратегії управління для ЦОД (або кожного його кластера) з урахуванням SLA, QoS, балансування навантаження, контролю доступу. Контроль доступу включає: швидкість відкидання запитів та швидкість надходження запитів.
Стадія 1. Рівень ЦОД. Вирішення оптимізаційної задачі, планування реалізації стратегії.	Взаємодія з програмно-визначеними підсистемами. Управління міграціями ВМ, робочими процесами і посередниками (middlebox) між кластерами і між федеративними ЦОД. Управління мережею і адресацією, перетворення адрес. Планування роботи з ВМ, підзадачі: - *пошук ФС для запуску ВМ*; - *пошук ФС для міграцій ВМ*.
Стадія 2. Рівень ГМ. Реалізація стратегії, вирішення завдань управління.	Управління робочими процесами і посередниками, підзадачі: - *моніторинг роботи сервісів;* - *створення/зупинення сервісів;* - *управління взаємодією сервісів;* - *розміщення паралельно працюючих екземплярів сервісу на різних ФС*. Диспетчеризація роботи ФС, підзадачі: - *збір даних з ФС;* - *включення/вимикання ФС;* - *включення/вимикання комутаторів стійки (ToR);* - *консолідація ВМ.* Визначення політик управління для рівня ФС при сталому режимі роботи ЦОД. Облік якісних показників роботи кластера та відправка якісних показників роботи кластера в ГМ ЦОД (при наявності кластерів ФС).
Рівень ФС. Менеджер фізичного сервера.	Моніторинг і управління ВМ. Міграція ВМ. Прогнозування навантаження на кожну ВМ. Вертикальне масштабування ВМ.

Важливе значення при управління хмарним ЦОД мають механізми і методи управління, які вбудовані в існуючі системи управління і оркестрації хмарних

послуг і називаються політиками. Політики дозволяють додавання нових керуючих впливів, які активуються в результаті виконання простих умов у визначених станах системи. Політики управління – це послідовність елементарних керуючих впливів, не пов'язаних з вирішенням оптимізаційної задачі, які потрібно застосувати у відповідь на виникнення визначених вимог. Окрема увага приділяється визначенню політик управління на рівні ФС при сталому режимі роботи ЦОД. Політики управління призначені для підтримання сталого режиму роботи ЦОД при стаціонарному навантаженні, коли всі ВМ працюють в межах виділених ресурсів і виникає необхідність в керуванні окремими ВМ або сервісами, в яких, наприклад, відбуваються зміни кількості клієнтських запитів (зменшення або збільшення). Для такого сценарію спрацьовує політика видалення або створення нової ВМ в тій самій стійці ФС з метою зменшення міжстійкової (або міжкластерної) мережевої взаємодії. Інші політики можуть бути такі: визначення ФС для переведення в сплячий режим, вибір ВМ для міграції, вибір ФС для розміщення нової ВМ або міграції існуючої. Таким чином, політики можливо застосувати при надлишку ресурсів на протязі певного часу роботи ЦОД в сталому режимі. Сталий режим визначається за допомогою метрик миттєвого і середнього коефіцієнтів життєздатності ВМ (3.23), розроблених у розділі 3.

IaaS є базою і основною сервісною моделлю в сучасних хмарних ЦОД, яка надає широкий спектр послуг від віртуальних робочих столів (VDI) до послуг більш високого рівня, таких як, PaaS і SaaS. Нині перспективний напрямок розвитку – це сервіси, що засновані на моделях PaaS і SaaS. Про це свідчить білінгові політики провайдерів хмарних послуг. Кінцевому користувачеві вигідніше використовувати існуючі хмарні сервіси в хмарі моделі SaaS, замість того, щоб розгортати свою власну службу на виділеному віртуальному сервері.

Надання додаткових ресурсів здійснюється за наявністю вільних ресурсів. Запити клієнтів на збільшення обсягів наданих ресурсів повинні задовольнятися без зниження якості сервісів для інших клієнтів. При цьому власник ЦОД завжди прагне уникнути надмірного виділення ресурсів, які зараз використовуються в поточний час.

Існуюча практика резервування ресурсів з запасом для того, щоб попередити небажані наслідки, викликані браком ресурсів при зростаючих потребах клієнтів і зростанні робочого навантаження, призводить до збільшення експлуатаційних витрат і витрат на електроенергію. Тому в СУІ слід використовувати методи і алгоритми, що дозволяють скоротити обсяги зарезервованих ресурсів без зниження якості послуг, ІТ-сервісів і виключення випадків порушення SLA.

2.5.3. Моделі системи управління ресурсами ІТ-інфраструктури

Для забезпечення ефективного використання ресурсів хмарних ЦОД, необхідного рівня сервісу, масштабування і адаптації до розгортання нових послуг, пропонується структурно-функціональна модель типової програмно-визначеної системи управління ІТ-інфраструктурою хмарного ЦОД, яка показана на рис. 2.4.

Рис. 2.4. Структурно-функціональна модель типової програмно-визначеної системи управління ІТ-інфраструктурою хмарного ЦОД

Запропонована модель складається з трьох шарів і розроблена з урахуванням сучасних підходів до управління ресурсами ЦОД, принципів використання програмно-визначених технологій, віртуалізації і політик

управління. Склад шарів архітектури вибирається на підставі можливості реалізації централізованого управління, а також децентралізованого управління з координацією.

Перший шар являє собою шар інфраструктури і складається з площини фізичного обладнання і з площини віртуалізації. Фізична площина складається з фізичних ресурсів ЦОД, таких як сервери, мережеві стелажі, сховища, комутатори стелажу і довільна топологія мережі. Фізична площина в сучасних ЦОД може бути складена з фізичних контейнерів [129].

Такий тип ресурсів являє собою окремі приміщення, що складаються з ряду стелажів, в яких розміщуються ФС. Площина віртуалізації являє собою віртуальну інфраструктуру, що визначена користувачем, і складається з ВМ, контейнерів, які забезпечують надання IT-послуг, і виконання робочих процесів. Об'єкти площини віртуалізації управляються за допомогою гіпервізора або іншого програмного забезпечення низького рівня, що встановлене на ФС. Об'єкти площини віртуалізації пов'язані з один з одним віртуальними шляхами і взаємодіють під управлінням контролерів програмно-визначеного шару.

Другий шар, програмно-визначений, являє собою шар для існуючих програмно-визначених контролерів, використовуваних для роботи в мережі, для зберігання даних, та для управління обчисленнями в науково-дослідних або виробничих умовах [282]. Наприклад, POX [253], NOX [254], ONOS [255], OpenDaylight [256] – це контролери на основі протоколу OpenFlow для реалізації SDN [257]. Функції програмно-визначеного контролера сховища, як правило, інтегровані в ту чи іншу файлову систему з підтримкою централізованої або розподіленої файлової системи з блоковим або файловим принципом зберігання. У п. 5.8 розроблений метод управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД, який реалізований у програмно-визначеному контролері сховища. На цьому рівні контролери розроблені і випробувані незалежно від розроблення і модифікації шару рівня оркестрації в системі управління. Програмне забезпечення програмно-визначеного контролера передбачає наявність відкритого API, щоб контролювати і управляти процесами шару інфраструктури за допомогою підсистем шару оркестрації.

Шар оркестрації забезпечує зовнішній інтерфейс між кінцевим користувачем і ресурсами хмарного провайдера, включаючи різні *політики управління* (management policies), такі як відповідність SLA, балансування навантаження, цінова політика, відповідність рівню енергоспоживання, забезпечення рівня продуктивності і рівня QoS. На цьому шарі реалізована система управління IT-інфраструктурою хмарного ЦОД через диспетчерування площини фізичного обладнання, визначення стратегії управління і формування керуючих впливів. На шарі оркестрації реалізовані також політики управління мережею, зберіганням даних і обчисленнями. Програмно-визначений шар забезпечує механізми виконання цих політик а площина фізичного обладнання виконує його.

На шарі оркестрації запити користувача надходять в модуль управління допуском до сервісу, який приймає рішення про те, чи можливо обробити запит, чи ні. Рішення базується на політиці, отриманої з модуля вибору політики управління, включаючи SLA, перевірку особистості, дозволи доступу, оплачені послуги та інші. Модуль визначення стратегій управління реалізує інтегрований метод керування, який полягає у адаптації через відповідний вибір стратегій, методів і політик управління залежно від поточного і прогнозованого стану ФС хмарного ЦОД на основі тенденцій використання ресурсів і розроблених метрик (миттєвий і середній коефіцієнти життєздатності ВМ, індикатор дисбалансу ФС, коефіцієнт відношення необхідних ресурсів до середнього об’єму наявних ресурсів та поріг вільних ресурсів), які розроблені в розділах 3 і 5. Прогнозування навантаження на ФС виконується менеджером фізичного сервера (МФС), використовуючи адаптивний метод комбінованого прогнозування навантаження на обчислювальні ресурси хмарного ЦОД, який розроблений у розділі 4.

Модуль планування відповідає за розгортання хмарних ресурсів на вимогу клієнта і реалізує метод управління ємністю ЦОД, розроблений в п. 5.7, який використовує розроблені метрики, враховує гетерогенність ФС і визначає тип і кількість ФС, що треба увімкнути для обслуговування прогнозованого навантаження. Він отримує вимоги, відповідні політики і рівень поточних запитів для генерації пропозиції модулю управління ЦОД для розміщення

ресурсів на фізичному плані за допомогою програмно-визначених контролерів. Модуль управління ЦОД реалізує основні алгоритми для визначення ФС, де буде розгорнуто ВМ користувача, віртуальний допоміжний сервер (middlebox) або робочий процес (workflow). Цей модуль управління також отримує інформацію про віртуальні маршрути, що з'єднують ВМ, віртуальні допоміжні сервери і сервери з робочими процесами. Модуль управління ЦОД генерує відповідні команди і відправляє їх до програмно-визначених контролерів, використовуючи API або відкриті протоколи. Модуль управління ЦОД реалізує інтегрований метод керування фізичними і віртуальними машинами в IT-інфраструктурі хмарного ЦОД на основі динамічної моделі станів, із застосуванням методів стохастичного пошуку для виконання міграцій ВМ, вивільнення, увімкнення і вимкнення ФС та прогнозування для адаптації до змін навантаження.

Схема реалізації стратегій управління IT-інфраструктурою хмарного ЦОД в типовій програмно-визначеній СУІ представлена на рис. 2.5 [307].

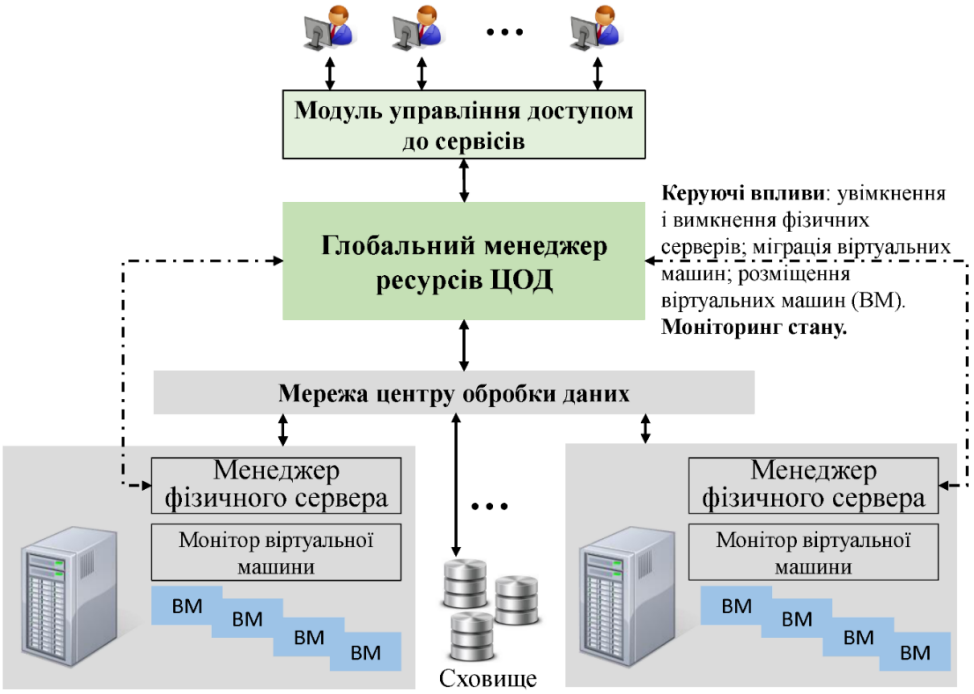


Рис. 2.5. Схема реалізації стратегій управління хмарним ЦОД

Структура об'єкта управління представлена множиною ФС, кожен з яких характеризується гетерогенним апаратним забезпеченням і операційною платформою. Крім того, кожен ФС має фіксовану потужність ресурсів. ФС надає ресурси для створення роботи кількох ВМ. ФС підключаються в єдину мережу

ЦОД за допомогою певної мережевої топології. У центрі оброблення даних кількість ВМ завжди змінюється. Необхідно враховувати також, що в динамічному середовищі деякі ВМ, визначені для міграції, можуть припинити існування під час визначення керуючих впливів.

Глобальний менеджер (ГМ) є основним модулем централізованого управління обчислювальними ресурсами ЦОД і реалізується як відмовостійкий і продуктивний кластер ФС [16]. ГМ реалізує такі запропоновані в дисертації методи управління: метод рівномірної консолідації ВМ з використанням модифікованого методу імітації відпалу, двостадійний метод управління ресурсами хмарного ЦОД на основі модифікованого методу променевого пошуку, метод динамічної консолідації і розміщення ВМ на основі модифікованого методу навчання з підкріпленням, метод управління міграцією та розміщенням ВМ в сталому режимі, метод управління потужністю ЦОД. ГМ реалізує адаптивний програмно-визначений підхід через управління ресурсами хмарного ЦОД, здійснюючи керуючі впливи на віртуальні і фізичні ресурси ЦОД в умовах програмно-визначеного середовища, і дозволяє вибирати стратегії, методи і політики управління ІТ-інфраструктурою з метою адаптації до впливу зовнішніх факторів.

На прикладному рівні існує набір клієнтів, які запитують послуги або замовляють виконання пакетних завдань, що вимагають створення однієї або декількох ВМ. Кожна ВМ характеризується конкретними показниками продуктивності, зазначеними в угоді SLA. Модуль управління доступом до сервісів відповідає за обробку запитів користувачів таким чином, щоб задовольнялися вимоги SLA. Одна ВМ обробляє одне пакетне завдання одночасно.

Схема реалізації управління ІТ-інфраструктурою хмарного ЦОД на базі розробленого у розділі 3 методу інтегрованого управління ресурсами (ІУР) показана на рис. 2.6. Метод ІУР вирішує такі задачі: консолідація ВМ, розміщення нової ВМ та задача планування ресурсної місткості. Запропонований метод інтегрованого управління включає в себе моделі енергоспоживання,

порушення умов SLA та управління ємністю ЦОД, що використовують модель динаміки ЦОД [284].

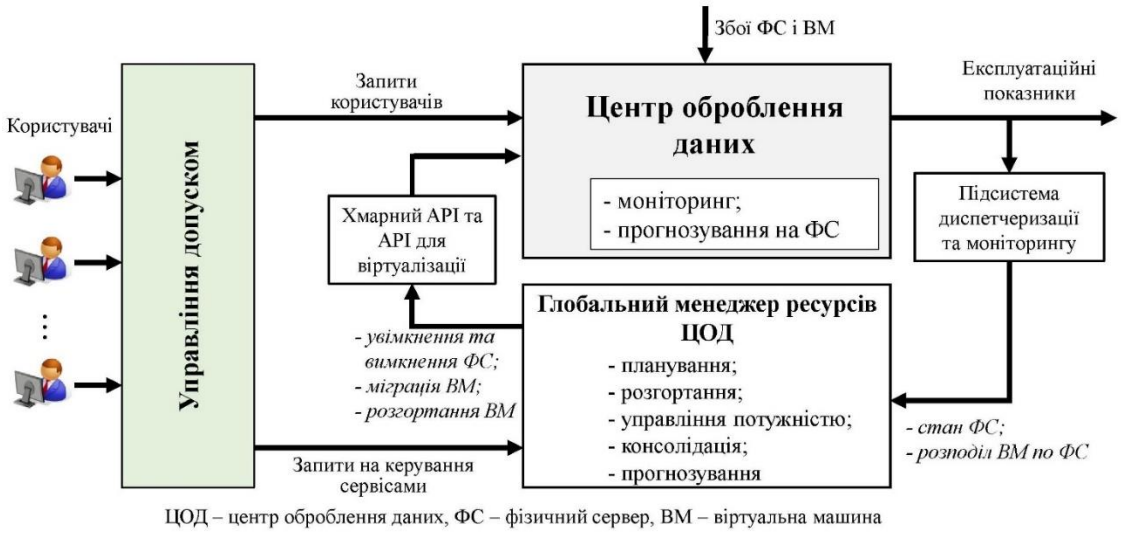


Рис. 2.6. Схема реалізації методу інтегрованого управління ресурсами хмарного ЦОД

Таким чином, методологія і відповідна концепція управління IT-інфраструктурою ЦОД провайдера хмарних послуг орієнтована на створення програмно-визначеного середовища інтегрованого управління IT-процесами, розподіленими застосунками та ресурсами хмарного ЦОД для трьох сервісних моделей хмарних обчислень. Концепція управління IT-інфраструктурою хмарного ЦОД як система шляхів і методів вирішення поставленої в роботі проблеми об’єднує такі положення та принципи: урахування програмно-визначених контролерів управління трьома основними ресурсами ЦОД (обчислення, сховище, мережа); застосування операторного підходу із визначенням і застосуванням стратегій управління ресурсами і навантаження хмарного ЦОД; декомпозиція хмарної IT-інфраструктури на три рівні (інфраструктури, платформи та застосунків); врахування традиційної, конвергентної і гіперконвергентної архітектур сучасних хмарних ЦОД; застосування адаптивного методу комбінованого прогнозування навантаження для визначення керуючих впливів на IT-інфраструктуру хмарного ЦОД; застосування методу інтегрованого управління гетерогенними ресурсами ЦОД з урахуванням порушень SLA, споживання електроенергії і необхідної потужності на наступному кроці управління; застосування методів стохастичного

локального пошуку (променевий пошук, відпал, навчання з підкріпленням) для схем реалізації визначених стратегій управління IT-інфраструктурою хмарного ЦОД; застосування методу управління розподіленим дворівневим сховищем з реплікацією для гіперконвергентних систем; врахування нових метрик стану IT-інфраструктури (миттєвий і середній коефіцієнти життєздатності ВМ, індикатор дисбалансу ФС, коефіцієнт відношення необхідних ресурсів до середнього об'єму наявних ресурсів, поріг вільних ресурсів) для визначення поточної стратегії управління; використання комбінації централізованого управління за допомогою глобального менеджера ЦОД і децентралізованого управління на рівні ФС залежно від обраної стратегії управління на поточний момент часу.

Концепція управління IT-інфраструктурою ЦОД провайдера хмарних послуг об'єднує запропоновані в роботі методи і алгоритми, а також існуючі принципи і технології.

2.6. Використання інтегрованих рішень для реалізації гіперконвергентних систем

Архітектура ЦОД зазнала значних трансформацій за останні два десятиріччя, від власних фірмових рішень до складних трирівневих систем [156]. Нині набуває поширення такий підхід до побудови ЦОД, при якому IT інфраструктура будується із залученням компонентів, модулів, програм і ліцензій від різних вендорів (рис. 2.7).

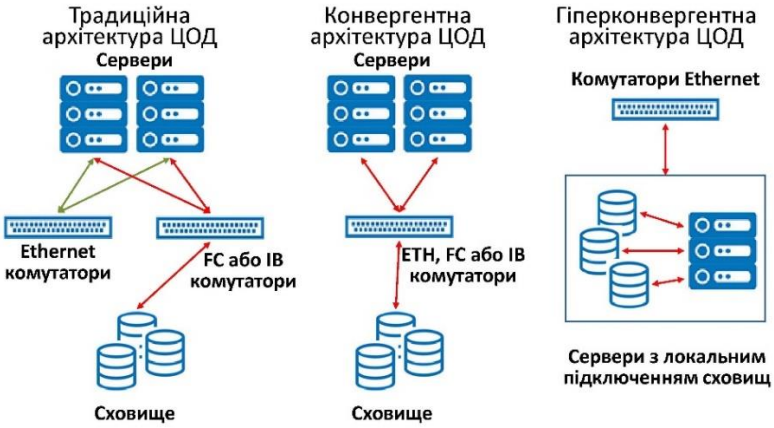


Рис. 2.7. Сучасні архітектури ЦОД

Один з найскладніших планів розгортання інфраструктури ЦОД включає в себе розроблення архітектури на основі існуючих специфікацій, аналіз рішень і вибір вендорів, придбання серверного обладнання, мережевих пристроїв, систем зберігання SAN і/або NAS, ПЗ і ліцензій, налаштування взаємодії обладнання та засобів управління. Якщо такий план розгортання виконує інтегратор, то підприємство несе ще більші витрати коштів. Якщо такий план розгортання підприємство виконує самостійно, то збільшуються витрати часу в цілому і витрати людино-годин. Розгортання та розміщення такої складної IT-інфраструктури вимагає значних капітальних та операційних інвестицій, а також перевірок сумісності. Крім того, прийняття рішення про розгортання SAN може призвести до так званої проблеми блокування постачальника (англ. vendor lock-in) і неможливості неруйнівного горизонтального розширення по шарах ЦОД.

Знизити витрати на проектування і розгортання ЦОД дозволяє інший підхід, який полягає в застосуванні інтегрованих рішень від певного вендора. Незважаючи на виникнення проблеми блокування постачальника підприємство отримує конкурентоспроможне рішення для ЦОД, яке вендор супроводжує, оновлює і усуває помилки інтеграції. Інтегрована система являє собою набір засобів, що включає певну кількість серверів, систем зберігання та мережевих пристроїв, які підібрані вендором (або групою вендорів) таким чином, щоб ефективно реалізовувати функції управління за допомогою спеціалізованого ПЗ, розробленого для цього класу інтегрованих систем. Розрізняють три категорії інтегрованих систем [140, 148, 162]: інтегрована система стека (англ. integrated stack system), інтегрована система інфраструктури (англ. integrated infrastructure system), гіперконвергентна система (англ. hyperconverged integrated system).

Інтегрована система стека, як правило, базується на блейд-системах і призначена для створення IT-інфраструктури для певного програмного стека, наприклад IBM PureApplication System, Oracle Exadata Database Machine and Teradata і ін. Така система складається з сервера, сховища і мережевого пристрою, як правило у вигляді різних модулів, і характеризується сильною «прив'язкою» до вендору.

Інтегрована система інфраструктури на сьогоднішній день набула найширшого розповсюдження в ЦОД малих і середніх розмірів, а також як окремі рішення в складі IT-інфраструктури підприємств і організацій завдяки доступності і простоті в управлінні. Такі системи, також, отримали назву конвергентні (англ. converged systems). Конвергентні системи представлені і у вигляді блейд формфактору і у вигляді стандартного 19-ти дюймового формфактору для стійки. Прикладом конвергентних систем є VCE Vblock, HP ConvergedSystem, CISCO UCS, Lenovo Converged System та інші. Така система складається з сервера, сховища і мережевого пристрою у вигляді різних модулів, і характеризується істотною «прив'язкою» до вендору або групи вендорів.

Основними перевагами конвергентних систем є: мінімізація часу простою всієї системи при оновленні ОС, програм і драйверів за рахунок застосування єдиного пакету оновлення (англ. roll-up); адаптація апаратних засобів для певного ПЗ і навантажень клієнта призводить до підвищення продуктивності роботи застосунків; можливість нарощувати апаратні потужності (scale-up) і користуватися супроводом вендора на тривалий термін.

Гіперконвергентна система побудована на основі web-scale архітектурної моделі, є розподіленою за своєю природою і складається з будівельних блоків (nodes), кожен з яких включає в себе обчислювальні (1-2 процесора) ресурси архітектури x86 (в більшості випадків), ресурси зберігання (HDD, SSD) і мережевої взаємодії (2-5 інтерфейсів Ethernet). Ресурси сховища представлені на стороні сервера компонентами NVMe PCI або DIMM, SSD і HDD. Мережева комунікація представлена декількома інтерфейсами з 10Гб/с і 40Гб/с Ethernet.

З використанням таких будівельних блоків пропонується побудувати високопродуктивний кластер, який дозволяє розгорнути, ефективно використовувати і масштабувати широкий клас застосунків, включаючи інфраструктуру BPC, сервіси для оброблення великих даних (в тому числі за схемою mapreduce), розподілені бази даних (SQL і NoSQL), файлові сховища і об'єднані комунікації. Основною відмінністю від конвергентних систем, серед інших важливих відмінностей, є відмова у більшості конфігурацій (але не виняткова) від централізованого сховища даних і мереж SAN/NAS, що дозволяє

істотно знизити капітальні та операційні витрати на розгортання, управління і масштабування сховищ. Значних результатів у виробництві гіперконвергентних систем досягли компанії Nutanix, VMware, HPE, CISCO, Microsoft, Scale Computing. Більшість інтегрованих систем використовують процесори Intel і AMD x86, а також в невеликих кількостях представлені архітектури RISC, Power, SPARC, ARM і Intel Atom.

Конкретні підходи і технології на основі гіперконвергентної архітектури розроблені у розділі 7.

Висновки до розділу 2

1. Розроблено архітектуру ІТУ ІТ-інфраструктурою хмарного ЦОД, яка спирається на узагальнену схему ІТ-інфраструктури хмарного ЦОД і використовує узагальнену схему реалізації функцій управління. При цьому враховуються суттєві характеристики хмарних обчислень, ієрархічність програмно-апаратної системи ЦОД і декомпозиція функцій управління. ІТУ включає в себе: підсистему диспетчеризації, моніторингу і прогнозування; підсистему визначення стратегій управління і вибору методів управління; підсистему генерації керуючих впливів, що впливають на програмно-апаратні засоби ЦОД.

2. Розроблено методологію управління ІТ-інфраструктурою хмарного ЦОД з урахуванням запропонованих в дисертації стратегічного рівня управління, ієрархічного управління, адаптації до поточного стану системи, поєднанням управління ІТ-інфраструктурою по стану з управлінням традиційними методами математичного програмування з використанням прогнозування. Розроблена методологія дозволяє вирішувати задачі управління ННДОЗС і досягти заданої якості надання хмарних послуг в ієрархічній структурі ІТ-процесів з використанням достатньої кількості ресурсів рівня ІТ-інфраструктури та адаптуватися до змін навантаження через прогнозування споживання цих ресурсів.

3. Розроблено операторну форму постановки аналізу і розв’язання задач управління ІТ-інфраструктурою, яка дозволяє будувати нові моделі хмарного

ЦОД і застосовувати різні методи і алгоритми управління у вигляді схем реалізації з метою адаптації до поточних умов функціонування. При цьому, в залежності від поточного стану хмарного ЦОД і визначеної стратегії управління, склад критеріїв і обмежень змінюються під впливом технологічних особливостей хмарного ЦОД та необхідності надання сервісів із зазначеними показниками якості.

4. Розроблено стратегії управління IT-інфраструктурою ЦОД провайдера хмарних послуг, які враховують особливості IT-інфраструктури хмарного ЦОД, суттєві характеристики хмарних обчислень, наявність програмно-визначених технологій в підсистемах управління обчисленнями, сховищами і мережею ЦОД, ієрархічність процесів управління IT-ресурсами і IT-послугами, невизначеність навантажень, динаміку процесів в хмарному ЦОД, гетерогенність апаратно-програмного забезпечення. З метою автоматичного визначення стратегії управління розроблено модель і метод нечіткого виводу, а також модель "гри з природою", які дозволять визначити стратегію управління в залежності від метрик оцінки стану хмарного ЦОД.

5. Розроблено структурно-функціональну модель типової програмно-визначеної системи управління IT-інфраструктурою хмарного ЦОД, яка складається з шару інфраструктури (апаратних засобів і системного ПЗ), програмно-визначеного шару (контролери управління гіпервізорами, сховищами і мережею) і шару оркестрації (підсистеми і модулі управління СУІ). У складі моделі розроблена схема реалізації стратегій управління IT-інфраструктурою в загальній СУІ та схеми реалізації методів управління ресурсами хмарного ЦОД.

6. Для вирішення основної проблеми розміщення та міграції ВМ ЦОД запропоновано адаптивний програмно-визначений підхід до розподілу ВМ хмарного ЦОД, який полягає в управлінні фізичними і віртуальними ресурсами через вибір стратегій управління з метою адаптації до зовнішніх впливів. Адаптивний програмно-визначений підхід реалізується глобальним менеджером, який управляє ресурсами ЦОД в сталому режимі і при високій інтенсивності створення/видалення ВМ, використовує оптимізаційні методи і комбінації евристик, які враховують дотримання заданих показників виконання

SLA, поточну завантаженість ресурсів ФС та комутаторів стійки ToR, поточну кількість міграцій ВМ, взаємні перешкоди між ВМ на одному ФС, кількість ВМ з пульсуючими робочими навантаженнями на окремому ФС. ГМ дозволяє вибирати різні стратегії, алгоритми і політики для управління ресурсами і ВМ з метою адаптації до впливу зовнішніх факторів, таких як зміна робочого навантаження, інсталяція оновлення, використання нових програмних і апаратних платформ, зміни в структурі сервісів та їх продуктивності провайдера послуг.

7. Визначено задачі управління ІТ-інфраструктурою хмарного ЦОД на різних рівнях управління з урахуванням впливу складу і вимог до надання ІТ-послуг, гетерогенності апаратно-програмних платформ в складі ЦОД, сервісних моделей надання хмарних послуг, використовуваних файлових систем і програмно-визначених контролерів. Розроблені схеми реалізації стратегій управління при нестачі або надлишку ресурсів, а також при стаціонарному або нестаціонарному навантаженні. Зазначено задачі, виконання яких реалізує обрану стратегію. Визначено, що при сталому режимі роботи ЦОД доцільно використовувати політики управління, в тому числі ті, що вбудовані в існуючі програмні засоби реалізації хмарних технологій.

8. Обґрунтовано необхідність використання інтегрованих рішень і гіперконвергентних систем в хмарних ЦОД, які дозволяють знизити витрати на проектування і розгортання хмарного ЦОД, мінімізувати час простою всієї системи при оновленні системного ПЗ, програм і драйверів за рахунок застосування єдиного пакету оновлення, адаптувати апаратні засоби для певного ПЗ і навантажень клієнта, підвищити продуктивність роботи застосунків, забезпечити можливість гнучко нарощувати апаратні потужності.

РОЗДІЛ 3. МОДЕЛІ І МЕТОДИ КОМПЛЕКСУ СХЕМ РЕАЛІЗАЦІЇ СТРАТЕГІЙ УПРАВЛІННЯ РЕСУРСАМИ ІТ-ІНФРАСТРУКТУРИ ХМАРНОГО ЦЕНТРУ ОБРОБЛЕННЯ ДАНИХ

У розділі розроблено схему реалізації стратегій інтегрованого ієрархічного управління ресурсами ІТ-інфраструктури; структуру ієрархічної системи управління хмарним ЦОД на основі декомпозиції хмарної ІТ-інфраструктури на рівні інфраструктури, платформи та застосунків; розроблено архітектуру і метод управління сервісами в ієрархічній системі хмарного ЦОД; обґрунтовано необхідність інтегрованого управління ресурсами ІТ-інфраструктури хмарного ЦОД; розроблено модель динаміки хмарного ЦОД на основі простору станів та інтегровану модель, що враховує споживання електроенергії, порушення умов SLA і планування ресурсної потужності; сформульовано і вирішено оптимізаційну задачу перерозподілу ресурсів в ІТ-інфраструктурі в складі методу інтегрованого управління ресурсами (ІУР); розроблено схему дворівневої системи управління гіперконвергентною ІТ-інфраструктурою і підходу до адаптивного розміщення ВМ з використанням прогнозу навантаження.

3.1. Ієрархія процесів управління ІТ-інфраструктурою

У сучасних хмарних ЦОД кількість послуг та клієнтів значно зростає, також зростає складність програмного забезпечення та ІТ-інфраструктури. Сучасні виклики призводять до розроблення нових архітектур для управління ІТ-інфраструктурою ЦОД, адаптуючись до появи нових інформаційних послуг та нових вимог у процесі цифрової трансформації бізнесу. Усе це вимагає скорочення ручного управління, адміністрування та операцій налаштування. Замість цього провайдери хмарних сервісів повинні витрачати більше часу на створення інноваційних послуг, що підвищують конкурентоспроможність і прибутковість. У деяких випадках різні ІТ-команди постачальника ІТ-послуг керують відокремленими підсистемами, такими як налаштування, управління та підтримка гіпервізора, сховища або мережі. Але в сучасних хмарних ЦОД ці

функції повинні виконуватися через API, використовуючи інструменти оркестрації та інтеграції.

Хмарні ЦОД – це складні системи, які використовують концепцію виконання обчислень на замовлення з метою забезпечення високопродуктивними IT-послугами користувачів і орендарів [59]. IT-інфраструктура хмарних ЦОД використовуються постачальниками послуг для забезпечення великого числа користувачів доступом до інформаційних послуг, застосунків, обчислювальних ресурсів на основі трьох моделей хмарних послуг: інфраструктура як послуга, платформа як послуга і програмне забезпечення як послуга [60]. Для ефективного управління хмарним ЦОД та надання інформаційних послуг на певному рівні якості необхідно автоматизувати роботу всіх підсистем базової IT-інфраструктури, забезпечуючи при цьому задану надійність і безпеку. При цьому значні зусилля та увага приділяються задоволенню потреб користувачів у доступі до обчислень, систем зберігання даних та застосунків, що надаються на вимогу з одночасним виконанням вимог SLA.

Для досягнення цих цілей використовуються два основні підходи, які зазвичай відносяться до централізованого управління або розподіленого управління. У підході на основі централізованого управління глобальний менеджер має хороший огляд системи і обчислює управляючі впливи щоб змінити конфігурацію IT-інфраструктури хмарного ЦОД. Однак архітектура централізованого контролера негативно впливає на масштабованість рішення і вимагає додаткових заходів для забезпечення відмовостійкості, підвищення продуктивності моніторингу і роботи мережі ЦОД. Підхід розподіленого управління добре масштабується, але має слабкі можливості для визначення та оптимізації стану системи в цілому.

У деяких дослідженнях ЦОД розглядається як сукупність кластерів, де кожен кластер складається з стійок, і кожна стійка складається з ФС [64, 65]. Ці кластери знаходяться під управлінням контролерів, розташованих на різних рівнях ієрархії системи управління. Крім того, кластер розглядається як одна з абстракцій у визначеній ієрархії. Ще одним типовим рівнем ієрархії,

представленим у деяких дослідженнях, є віртуалізація [81, 134, 82]. Технології віртуалізації дають можливість абстрагуватися від особливостей окремих груп фізичних ресурсів, об'єднавши їх в апаратні і програмні підгрупи бажаної конфігурації і спростивши їх управління і налаштування. Ресурси ЦОД можуть бути абстраговані з використанням рівня віртуалізації, що заснована на створенні ВМ або контейнера. Провайдери хмарних послуг пропонують широкий спектр конфігурацій ВМ для вирішення все більш різноманітних задач з різним навантаженням, різними вимогами до продуктивності та до витрат [84, 85, 86]. Таким чином, користувачі повинні вибрати правильну конфігурацію ВМ для оброблення заданого навантаження з метою реалізації бізнес-потреб. Крім того, існує потреба у задоволенні вимог користувача застосунків при керуванні віртуалізованою інфраструктурою.

Для того, щоб зменшити вплив складності хмарного ЦОД в дисертаційній роботі пропонується застосовувати теорію ієрархій [61, 62] через розкладання складної системи на рівні, що складаються з процесів, якими можна керувати окремо, беручи до уваги різні цілі. Вимоги до системи управління можуть накладати обмеження на отримані рівні ієрархії. Необхідно враховувати такі особливості хмарного ЦОД: кількість інформаційних процесів, керованих на кожному рівні в ієрархії, може збільшуватися з часом; управляючі впливи на програмне та апаратне забезпечення генерується різними користувачами та різними підсистемами управління; з’являються нові інформаційні процеси, якими необхідно керувати на кожному рівні в ієрархії; нове програмне забезпечення та обладнання на кожному рівні в ієрархії може бути розгорнуто з часом [293].

Декомпозиція процесів управління ІТ-інфраструктурою хмарного ЦОД стала звичайною практикою постачальників інформаційних і хмарних послуг. Але в сучасному інформаційно-комунікаційному світі декомпозиція має бути реалізована в більш глибинних рівнях абстракції. Тому в роботі використовується абстрактна декомпозиція ресурсів як перший ключовий аспект управління ЦОД, і запропонований підхід може бути реалізований в системах,

визначених програмно, таких як Software Defined Network, Software Defined Storage, NFV та інші системи.

Другий основний елемент запропонованого підходу стосується ресурсів ЦОД. Проте в даній роботі запропонована звичайна схема компенсації. Кожен елемент, компонент і підсистема виконують зумовлені завдання з власними ресурсами. У разі нестачі ресурсів кожна підсистема має можливість запитувати додаткові ресурси з верхнього рівня ієрархії.

Третя фундаментальна характеристика запропонованого підходу полягає в ефективному поєднанні вибору стратегії управління з відповідними інструментами її реалізації на всіх рівнях ієрархії. Вибір стратегії управління відбувається за умови надмірного забезпечення або недостатнього забезпечення ресурсів ЦОД. Впровадження підходу ґрунтується на сукупності математичних моделей та методів оптимального управління та контролю з систематичним використанням прогнозування навантаження та машинного навчання.

Четвертим основним елементом запропонованого підходу є комбіноване використання управління в просторі станів системи та моделей оптимізації на основі математичного програмування та дослідження операцій. Запропонований підхід можна використовувати як основу для впровадження гнучких і настроюваних систем для провайдерів хмарних послуг. Це дозволяє сформулювати задачу управління в єдиній операторній формі і вибрати відповідні моделі і методи для реалізації оператора. Операторна форма постановки аналізу і розв’язання задач управління IT-інфраструктурою розроблена в п. 2.2. Вибір відповідних моделей залежить від стану базової IT-інфраструктури та особливостей бізнес-процесів.

Таким чином, в дисертаційній роботі пропонується представляти IT-інфраструктуру хмарного ЦОД як ієрархічну систему, що дозволяє її еволюціонування і перероблення для поліпшення її якісних і кількісних характеристик.

Поділ IT-інфраструктури на множини апаратних і програмних засобів є одним з широко використовуваних підходів до побудови ієрархії інформаційної системи. Проте просте розділення такої складної системи на апаратний та

програмний рівні призводить до проблеми адекватного відображення функцій програмних послуг (застосунків) на апаратні ресурси. Ця проблема є особливо гострою, коли система розширюється додаванням нових об'єктів управління і збільшенням їх кількості. Останнім часом з'явився і використовується програмно-визначений підхід для вирішення цих проблем [83]. Таким чином, врахування програмно-визначеної парадигми при розробленні рівнів ієрархії IT-інфраструктури хмарного ЦОД призводить до збільшення керованості, можливостей інтеграції та масштабованості отриманої архітектури.

В роботі також досліджено підхід до ієрархічного управління службами хмарного ЦОД, що поєднує традиційні методи управління ресурсами з управлінням шаблонами та конфігураціями на вищих рівнях запропонованої ієрархічної моделі хмарного ЦОД. При цьому зроблено висновок, що для представлення хмарного ЦОД як багаторівневої ієрархічної системи управління необхідно виконати декомпозицію на основі моделей хмарних обчислень SaaS, PaaS і IaaS. За результатами декомпозиції запропоновано багаторівневу ієрархічну систему управління для керування інформаційними послугами хмарного ЦОД за критерієм мінімізації витрат і мінімізації відхилень параметрів якості послуг від заданих. Таким чином, метою управління IT-інфраструктурою ЦОД є виділення оптимальної кількості ресурсів для реалізації послуг на кожному рівні системи.

Основний внесок полягає в декомпозиції об'єктів та послуг хмарного ЦОД таким чином, щоб ефективно керувати послугами та застосунками на вищих рівнях, впливаючи на параметри продуктивності рівня інфраструктури та впливаючи на конфігурації та структури рівня платформи та прикладного рівня. Обґрунтовано гіпотезу, що ця декомпозиція призведе до більш ефективного управління ресурсами і послугами ЦОД, що реалізує публічну та гібридну моделі розгортання хмари [60].

3.2. Ієрархічна модель системи управління хмарним центром оброблення даних на основі декомпозиції

Дослідження процесів керування в складних ієрархічних системах представлено у працях [62, 69, 70, 71, 72, 73]. Складні об'єкти, з точки зору управління ними, в більшості випадків можуть бути представлені системами, що складаються з декількох підсистем. У той же час ці системи мають різні рівні ієрархії. У складних ієрархічних системах кожна підсистема, з одного боку, є автономною, а з іншого боку, піддається впливу з боку інших підсистем. Кожна підсистема упорядкована ієрархічно і вирішує внутрішнє завдання управління (або завдання прийняття рішень) з урахуванням впливу вхідних і вихідних даних.

Одним з етапів визначення підсистем в складній ієрархічній системі є декомпозиція системи. Розкладання складної системи має сенс, оскільки система має велику розмірність, велику кількість станів і наявність різних критеріїв оптимізації, притаманних кожній конкретній підсистемі, які потенційно можуть бути отримані унаслідок декомпозиції. У процесі декомпозиції важливим фактором, який необхідно враховувати, є подання системи як об'єкта управління. Крім того, структура ієрархічної системи повинна виявляти найбільш значущі характеристики об'єкта управління, такі як: об'єкт управління складається з взаємопов'язаних підсистем прийняття рішень; ці підсистеми розташовані ієрархічно [70]. У даному випадку кінцевою метою є розроблення багаторівневої ієрархічної системи управління складним об'єктом, представленим IT-інфраструктурою хмарного ЦОД.

При проектуванні ієрархічних систем управління необхідно враховувати також їх багатокритеріальний і асинхронний характер. Багатокритеріальність розглядається як наявність локальних критеріїв оптимальності в кожній підсистемі ієрархії і існування критерію оптимальності для системи в цілому. Асинхронність розглядається як можливість вибору керуючого впливу для кожної підсистеми незалежно від порядку подання впливів управління на інші підсистеми. У цьому випадку впливи управління генеруються як у самій підсистемі (самоврядування), так і в підсистемах, які є вищими в ієрархії.

Використання багаторівневого ієрархічного підходу для розроблення системи управління IT-інфраструктурою хмарного ЦОД обумовлено високою складністю програмно-апаратної платформи, гетерогенністю програмного і апаратного забезпечення, наявністю декількох видів послуг на одному рівні ієрархії, необхідністю забезпечення високої надійності та безпеки, різною швидкістю плину інформаційних процесів, необхідністю використання різних моделей управління на кожному рівні ієрархії.

Наступним кроком розроблення багаторівневої ієрархічної системи управління є отримання інформації про об'єкт управління (і його складові), синтез законів управління для кожної підсистеми і системи в цілому, генерація сигналів управління та їх подання в підсистеми.

Хмарний ЦОД є складною системою з заданою ієрархічною структурою. Це вимагає розроблення методів управління з урахуванням наявності багатокритеріальних задач і асинхронного характеру плину процесів у великих масштабах. Унаслідок аналізу характеристик, властивих хмарним обчисленням [60, 74], моделей послуг і моделей розгортання хмари, необхідно зробити висновок, що для представлення хмарного ЦОД як багаторівневої ієрархічної системи управління необхідно виконувати декомпозицію на основі сервісних моделей хмарних обчислень IaaS, PaaS, SaaS. Моделі розгортання не придатні для використання при декомпозиції ЦОД, оскільки однакові структури та рівні абстракції наявні в кожній з моделей розгортання. Але моделі розгортання можуть бути використані для опису та моделювання взаємодії між провайдерами хмарних послуг та між приватними та публічними хмарами.

У рамках прийнятої архітектури повинна бути забезпечена скоординована взаємодія між різними структурами та суміжними рівнями ієрархії. Наприклад, щоб взаємодіяти з окремим верхнім рівнем ієрархії з сусіднім нижчим рівнем можна розробити структури нижнього рівня (апаратні та/або програмного забезпечення), які впливатимуть на властивості та якісні показники роботи інформаційних процесів, а також хмарного ЦОД в цілому [321]. Крім того, враховуючи наявні впровадження підходу інфраструктура як код (англ. Infrastructure as Code) [75, 76, 77], система управління має можливість динамічно

змінювати структури певного рівня ієрархії, впливаючи на якість надання послуг. Відповідно до сервісних моделей хмарних обчислень, запропонованих NIST [60], пропонується розкладати ЦОД на три рівні: рівень інфраструктури (фізичний план і віртуальна площина), рівень платформи та рівень ПЗ, як показано на рис. 3.1.

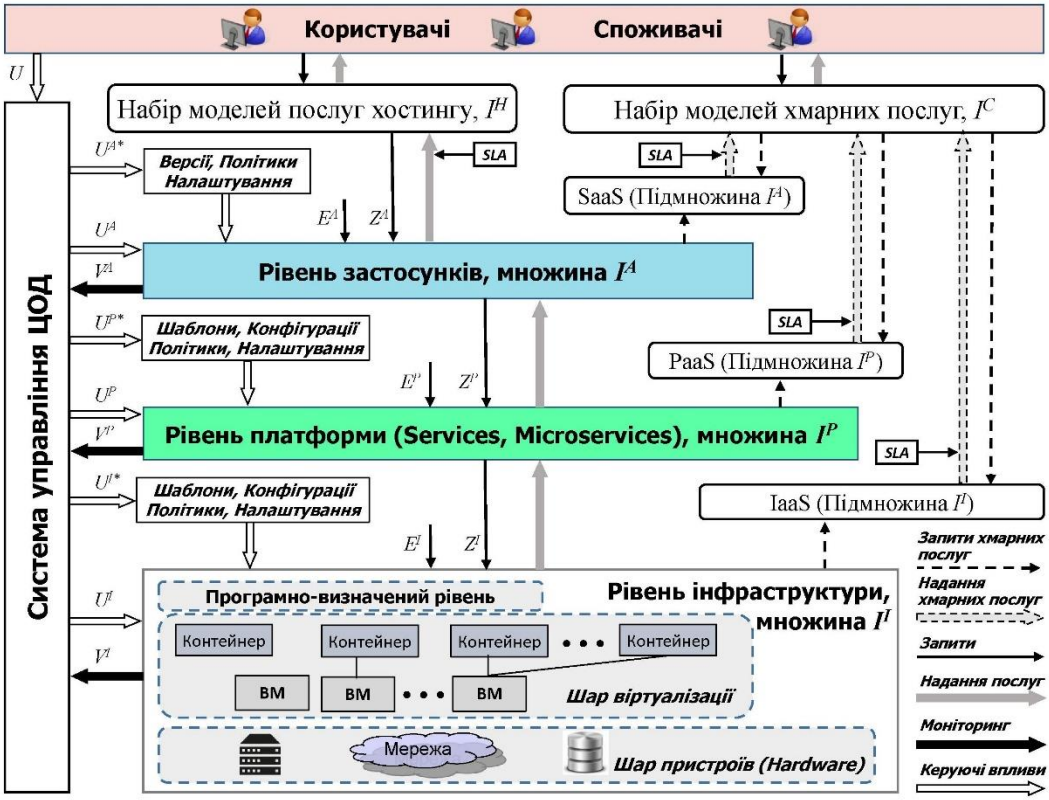


Рис. 3.1. Структурно-функціональна модель багаторівневої ієрархічної системи управління IT-інфраструктурою

У верхній частині над трьома рівнями представлені дві різні моделі споживання послуг: моделі хмарних сервісів і моделі послуг хостингу. Важливою відмінністю є те, що в моделях хмарних сервісів попередні завдання конфігурації та структурні зміни на кожному рівні виконуються постачальниками хмарних послуг. Провайдер хмарних послуг надає користувачам змогу створювати окремі інфраструктури I на кожному рівні.

3.2.1. Рівень інфраструктури

Рівень інфраструктури представлений набором інфраструктур I^I. Цей рівень є важливим компонентом при реалізації хмарних обчислень. Він відповідає за управління фізичними ресурсами, включаючи ФС, стійкові комутатори,

обладнання мережі та систем зберігання даних. ЦОД зазвичай містить сервери, які з'єднані між собою через комутатори стійки та інші мережеві системи [319, 329]. Рівень інфраструктури є основою для створення пулу віртуалізованих обчислювальних ресурсів, мереж і ресурсів зберігання даних через розбиття фізичних ресурсів за допомогою віртуалізації та програмно-визначених технологій. Наразі провайдери хмарних послуг пропонують різні конфігурації і типи ВМ [86, 84, 85], які налаштовані для обслуговування певного навантаження або виконання певного ПЗ. Наприклад, деякі розміри ВМ оптимізовані для інтенсивного використання процесора, пам'яті або системи збереження даних.

Рівень інфраструктури надає послуги за запитом з платформного рівня і вимагає впливу на конфігурації та визначення управління, викликаного процесами на рівні платформи. Функції управління, такі як оновлення, відображення віртуальних ресурсів на фізичні, підключення до мережі, забезпечення роботи СЗД виконуються провайдером, використовуючи керуючі впливи U^I. Функції управління, такі як конфігурація віртуального ресурсу, вибір шаблону об'єкта, визначення розмірів ВМ, використання мережі та СЗД, виконуються користувачем за допомогою керуючі впливи U^{I*}.

Рівень інфраструктури забезпечує надання ВМ та контейнерів. У кожній ВМ встановлена операційна система, а контейнер буде створено поверх базової операційної системи. Після того, як підмножина інфраструктурного рівня налаштована, запити користувачів рівня платформи, пов’язані з виконанням послуг та роботою прикладного ПЗ можуть бути реалізовані. Це може бути зроблено у відповідності з вимогами моделі хмарних сервісів або моделі послуг хостингу. Рівень автоматизації процесів конфігурації на кожному рівні ієрархії визначається постачальником послуг або операційними групами користувача (орендаря). Конфігурацію низького рівня мережевих з'єднань та сховищ повністю визначає постачальник інфраструктурних послуг.

У разі надання хмарних рішень IaaS, які є підмножиною інфраструктури провайдера, можна використовувати стандартизовані конфігурації, не вимагаючи від адміністраторів налаштовувати базове середовище виконання. Визначений користувачем набір елементів інфраструктури дозволяє користувачеві (або орендарю) розгортати і запускати довільні операційні

системи, платформи, фреймворки, служби та програми як з використанням набору моделей хмарних сервісів I^C, так і з використанням моделі послуг хостингу I^H. Таким чином, елементи інфраструктури з застосованими налаштуваннями і шаблонами являють собою об'єкт управління рівнем інфраструктури. Такі об'єкти і групи об'єктів належать до набору інфраструктур I^I і є основою для створення і забезпечення фунціонування об'єктів платформи.

Наприклад, база даних як послуга (DBaaS) та подібні пропозиції, як правило, враховуються як IaaS, незважаючи на те, що операційні групи не повинні в значній мірі налаштовувати базове середовище виконання тому, що воно не дуже корисне для кінцевих користувачів. Воно вважається частиною невидимої для користувача базової інфраструктури, і, як наслідок, зазвичай вважається, що DBaaS підпадає під визначення IaaS.

3.2.2. Рівень платформи

Об'єкт управління на рівні платформи представлений елементом з набору платформ I^P. Основна функція рівня платформи полягає в тому, щоб зробити екземпляр середовища для запуску сервісів і застосунків. Іншою метою рівня платформи є забезпечення роботи передбачуваного стандартного набору середовищ виконання та фреймворків. Наприклад, існує низка загальних платформ (таких як LAMP, WAMP, XAMPP [78], контейнерні послуги [79]), які забезпечують множину керованих середовищ виконання (наприклад, компоненти, проміжне програмне забезпечення та ОС) для користувачів [80]. Рівень платформи повністю управляється і регулярно оновлюється провайдером послу, використовуючи впливи управління U^P. Платформи, які пропонуються користувачам у вигляді хмарного сервісу, зазвичай використовуються для створення хмарних застосунків (англ. cloud-native applications).

Основною перевагою хмарних застосунків є розділення процедур конфігурування та збереження даних від середовища виконання застосунків. Прикладна функція, створена у хмарі, є повністю композиційною. Це дає змогу незалежно розгортати такі послуги, як мікросервіси. На відміну від монолітних архітектур розроблення застосунків мікросервісів можна запустити і співставити з необхідними бізнес-процесами за допомогою повністю автоматизованого

фреймворку розгортання і оркестрації. Таким чином, хмарний застосунок являє собою набір сервісів, кожен з яких працює в своєму власному процесі і взаємодіє з іншими сервісами на основі спрощених протоколів в асинхронному режимі. Мікросервіси добре підтримуються архітектурою контейнерів [79].

Якщо виявлено проблему, контейнери з хмарним застосунком видаляються і створюються нові контейнери з середовищами, до яких застосовуються код скриптів та конфігурування, а дані повторно приєднуються до нових примірників. Але коли апаратне та програмне забезпечення, що лежить в основі, повинно бути налаштоване певним чином, наприклад, для підвищення продуктивності вищерозташованих застосунків, то платформа як послуга не підходить.

Крім того, користувач рівня хмарної платформи не керує об'єктами, що відповідно працюють на рівні інфраструктури. Користувач керує розгорнутими застосунками та конфігураціями контейнерів, використовуючи впливи керування U^{P*}, такі як керування сеансами та контентом, інтеграцією пристрою та різними незалежними від платформи реєстрами для забезпечення функціонування застосунків. Рівень платформи надає послуги за запитом з прикладного рівня і вимагає налаштування та управління, викликаного процесами прикладного рівня.

Великі постачальники хмарних платформ широко впроваджують сумісні архітектури. Наприклад, контейнери Windows, контейнери Hyper-V і Docker контейнери – це чергова еволюція у віртуалізації. Контейнери завдяки гранулярності, ізольованості і керованості виступають основними ресурсами рівня платформи, що забезпечують портативні операційні середовища [158, 159]. Поряд з контейнерами Windows Server, Nano Server є ключовим елементом сучасних функцій прикладної платформи операційної системи Windows Server, готових до використання в програмно-визначеному середовищі [160].

3.2.3. Прикладний рівень

Об'єкт управління на прикладному рівні представлений елементом з набору застосунків і служб I^A. Основна функція прикладного рівня полягає в тому, щоб запропонувати застосунок, готовий до використання кінцевими користувачами.

У разі надання хмарних SaaS-рішень, які є підмножиною служб прикладного рівня, стандартизовані конфігурації можуть використовуватися без налаштування базового середовища на рівні платформи. Користувачеві доступні лише інструменти конфігурування на рівні застосунку (налаштування, політики та версії) в якості впливів U^{A*} для керування набором програм. Інші функції управління, такі як оновлення та конфігурування безпеки, виконуються провайдером послуг за допомогою впливу управління U^{A}. У такій архітектурі всі операції з конфігурування середовища виконання на рівні платформи виконуються провайдером або орендарем, які пропонують певну платформу.

Користувач може використовувати різні клієнтські пристрої, включаючи веб-браузери, для доступу до різних типів застосунків: безкоштовні хмарні програми, корпоративні послуги, програмне забезпечення для настільних комп'ютерів, програмне забезпечення для управління продажами, електронна пошта, сховища, засоби об’єднаних комунікацій та інші. Інший підхід користувачів до розгортання та використання власних застосунків у хмарі полягає у виконанні міграції з локального варіанту на хмарний варіант. У той же час існує багато обмежень і вимог, які повинні задовольнятися при переміщенні локальних застосунків до хмари [89, 90, 91]. Але модульна архітектура елементів на прикладному рівні дозволяє досягти кращої продуктивності, доступності та зниження експлуатаційних витрат за рахунок використання функції автоматичного масштабування на рівні платформи, що забезпечує роботу застосунків.

3.3. Модель інтегрованого управління сервісами хмарного центру оброблення даних в ієрархічній системі

Інформаційні процеси хмарного ЦОД розглянемо як складний об'єкт управління. Кожен інформаційний процес працює на одному або декількох рівнях (інфраструктура, платформа, застосунок), отримані унаслідок декомпозиції (п. 3.1). Мета кожного інформаційного процесу (або груп з них) у хмарному ЦОД полягає в наданні послуг з заданим рівнем якості. Таким чином, основною метою є розроблення багаторівневої ієрархічної системи управління для управління інформаційними процесами хмарного ЦОД через генерацію

керуючих впливів на кожному рівні за критерієм мінімізації витрат на підтримання ІТ-інфраструктури і мінімізації відхилень параметрів якості послуг від заданих.

Важко визначити централізований підхід до керування ІТ-інфраструктурою хмарного ЦОД за вищезазначеними критеріями. Причиною цього є складність ЦОД, описана в п. 3.2. Для досягнення основної мети пропонується підхід декомпозиції до ієрархічного управління інформаційними процесами хмарного ЦОД [328]. Задача керування якістю послуг, що надаються ІТ-інфраструктурою ЦОД, представлено як проблема прийняття рішень.

Для опису моделей і вирішення задач управління інформаційними процесами хмарного ЦОД введемо такі змінні:

A_i^h – це *i*-й сервіс, який надає користувачеві хмарний ЦОД на рівні *h*, $h \in H$, де *H* являє собою набір типів рівнів, таких як інфраструктура (*I*), платформа (*P*) і застосунок (*A*);

$R_i^{h,k}$ $i = \overline{1, M^h}$, $k \in K$ – це споживання ресурсів *i*-м сервісом на *h*-му модельному рівні, де M^h – це кількість сервісів на *h*-му рівні ЦОД, *K* – набір типів ресурсів, таких як процесор, пам'ять, швидкість вводу/виводу СЗД і т.д. Група ресурсів на рівні інфраструктури $R_i^{h,k}$ підтримує виконання *i*-го сервісу;

U_i^h – набір управлінських впливів, які керують *i*-ю послугою на *h*-му рівні ЦОД;

Q_i^h – це якість надання сервісу *i* на рівні *h*;

Q_i^{h*} – це задана якість надання сервісу *i* на рівні *h*;

Z_i^h – це запити *i*-го сервісу на *h*-му рівні моделі;

$q_{i,l}^h \in Q_i^h$, $l = \overline{1, L_i^h}$ – *l*-й індикатор якості *i*-го сервісу на *h*-му рівні ЦОД;

L_i^h – кількість якісних показників *i*-го сервісу на *h*-му рівні ЦОД;

V^h – сигнали зворотного зв'язку, отримані з *h*-го рівня в якості даних моніторингу;

E^h – збурюючі впливи, які впливають на *h*-й рівень ЦОД.

Якість надання *i*-го сервісу на рівні інфраструктури визначається як $Q_i^I = f(U_i^I, U_i^{I*}, R_i^{I,k})$. Якість надання *i*-го сервісу на рівні платформи

визначається як $Q_i^P = f(U_i^I, U_i^{I*}, U_i^P, U_i^{P*}, R_i^{I,k}, R_i^{P,k})$. Якість надання *i*-го сервісу на рівні застосунків визначається як $Q_i^A = f(U_i^I, U_i^{I*}, U_i^P, U_i^{P*}, U_i^A, U_i^{A*}, R_i^{I,k}, R_i^{P,k}, R_i^{A,k})$. Ресурси $R_i^{P,k}$ і $R_i^{A,k}$ також виділяються на рівні інфраструктури і використовуються для підтримки сервісів і застосунків на рівні платформи і застосунків відповідно. Наприклад, операційна система ФС обслуговує контейнер на рівні платформи і, також, споживає ресурси $R_i^{P,k}$, тому сервіс, що працює всередині контейнера, побічно споживає додаткові ресурси.

Метою впливів управління U_i^h є виділення оптимальної кількості ресурсів $R_i^{h,k}$ для роботи сервісу A_i^h на рівні *h*. Управляючі впливи, які контролюють сервіс *i* на рівні інфраструктури, визначаються як система $<U_i^I, U_i^{I*}>$. Управляючі впливи, які контролюють сервіс *i* на рівні платформи, визначаються як система $<U_i^I, U_i^{I*}, U_i^P, U_i^{P*}>$. Управляючі впливи, які контролюють послугу *i* на рівні застосунків, визначаються як система $<U_i^I, U_i^{I*}, U_i^P, U_i^{P*}, U_i^A, U_i^{A*}>$.

Управляючі впливи $<U_i^I, U_i^P, U_i^A>$ включають включення/вимикання об'єктів ІТ-інфраструктури, міграцію ВМ, балансування навантаження, горизонтальне масштабування, вертикальне масштабування, резервування, відновлення, реплікацію, оновлення об'єктів на всіх рівнях. Управляючі впливи $<U_i^{I*}, U_i^{P*}, U_i^{A*}>$ включають застосування конфігурацій, шаблонів, політик та параметрів за допомогою інструментів інфраструктури як коду.

Набір *U* сигналів управління, що впливають на всі три рівні на кожному кроці управління, може бути представлений у вигляді декартового добутку шести множин $U = U_i^I \times U_i^{I*} \times U_i^P \times U_i^{P*} \times U_i^A \times U_i^{A*}$.

СУІ ЦОД отримує сигнали зворотного зв'язку V^I, V^P, V^A у вигляді даних моніторингу, які функціонально залежать від управляючих впливів *U*, входів *Z* і збурюючих впливів *E*.

СУІ ЦОД управляє інформаційними процесами (послугами) на кожному рівні *h* таким чином, що різниця між фактичними значеннями $q_{i,k}^h \in Q_i^h$ *i*-го показника якості надання послуг і заданими значеннями $q_{i,k}^{h*} \in Q_i^{h*}$ є мінімальною.

$$\sum_{i=1}^{M^h}\sum_{k=1}^{L_i^h}\left(q_{i,k}^h - q_{i,k}^{h*}\right)^2 \to \min, \forall h \in H \quad (3.1)$$

Ефективність управління сервісами може оцінюватися за якістю наданих послуг (3.1), за витратами на використання ресурсів ЦОД та за витратами (штрафами) порушення SLA. Вартість штрафу за порушення SLA P^{SLA} може бути оцінена у такий спосіб:

$$P^{SLA} = p_1 P^Q + p_2 P^d, \quad (3.2)$$

де P^Q – штраф за відхилення параметрів якості обслуговування від заздалегідь визначених, P^d – штраф за затримку розгортання ресурсу, $p_1 > 0$ і $p_2 > 0$ вагові константи. Витрати на пеню P^Q і P^d можуть розраховуватися щогодини з урахуванням кількості порушень (відхилень).

В дисертаційній роботі пропонується визначити експлуатаційні витрати на використання ресурсів ЦОД як сумарні експлуатаційні витрати у термінах енергоспоживання ЦОД у такий спосіб:

$$P^E = e(E^I + E^U), \quad (3.3)$$

де E^I – загальне споживання електроенергії всім працюючим обладнанням на рівні IT-інфраструктури, а E^U – загальне споживання електроенергії, викликане управляючими впливами, e – ціна електроенергії.

Загальне споживання енергії E^I може бути визначено як $E^I = E_{PM} + E_N + E_S + E_C$, де E_{PM} – загальне споживання електроенергії ФС [322], E_N – загальне споживання електроенергії всім мережевим обладнанням, E_S – загальне споживання електроенергії всім обладнанням СЗД, а E_C – загальне споживання електроенергії всім обладнанням для охолодження.

Загальне споживання енергії E^U можна визначити як $E^U = E_{mVM} + E_M$, де E_{mVM} – загальне споживання електроенергії на міграцію ВМ [322], E_M – загальне споживання електроенергії для застосування всіх інших впливів управління, таких як оновлення, обслуговування, міграція даних, оптимізація мережевого трафіку, шифрування, балансування навантаження, реплікації та ін.

Загальною метою підходу декомпозиції до ієрархічного управління інформаційними процесами ЦОД є створення таких впливів управління *U*, які мінімізують відхилення параметрів якості обслуговування від заданих та мінімізують витрати на використання ресурсів ЦОД (3.3) та мінімізують штрафні витрати за порушення SLA (3.2). Таким чином, функція витрат (3.4) повинна бути мінімізована:

$$J = \alpha \left| P^{SLA} \right|^2 + \beta \left| P^E \right|^2, \tag{3.4}$$

де α і β - вагові коефіцієнти, визначені провайдером ЦОД, що позначають відповідно відносну важливість P^{SLA} і P^E.

Управління ресурсами хмарного ЦОД в цілому є нелінійною задачею цілочисельного програмування (3.4) і вирішується в дисертації на рівні інфраструктури. Оптимальне рішення можна знайти за допомогою класичних методів оптимізації. Але враховуючи велику кількість вхідних/вихідних змінних і обмежень, класичні методи оптимізації вимагають значних обчислень в реальному режимі часу для виконання оптимізації.

3.4. Метод інтегрованого управління ресурсами IT-інфраструктури

Результати досліджень ринку хмарних обчислень провідними консалтинговими та дослідницькими компаніями демонструють значне зростання кількості і якості хмарних послуг за останні десять років [103, 124]. Значна увага з боку дослідників та розробників приділяється збільшенню ефективності управління ресурсами IT інфраструктури для сервісної моделі IaaS з метою забезпечення клієнтів якісними послугами на вимогу в реальному часі з дотриманням вимог згідно угоди щодо дотримання рівня сервісу SLA. Хмарні обчислення на базі моделі IaaS складають найбільшу частину ринку хмарних послуг [104].

Провайдери хмарних послуг пропонують користувачеві широкий спектр сервісів, якість і продуктивність роботи яких напряму залежить від IT інфраструктури. Основними високорівневими сервісами є хмарні застосування, застосування електронної комерції, загальні застосування для бізнесу,

застосування для автоматизації процесів підприємства, застосування для розробників, пакетні завдання з середовищами виконання та застосування для сфери Інтернету речей. Кожний сервіс має конкретні вимоги до виконання, такі як висока доступність, висока продуктивність, еластичність зміни необхідної ємності та масштабування, балансування навантаження, розподілене функціонування і взаємодія з мобільними клієнтами. При цьому кожний хмарний сервіс та відповідна IT-інфраструктура сприймають різні комбінації робочих навантажень, які є нестаціонарними і які складно прогнозувати. Хмарні сервіси також є масштабованими та стійкими до відмов інфраструктури. Таким чином, дуже важливо забезпечити виконання угоди щодо дотримання рівня сервісу SLA, в тому числі, мінімальну затримку при плануванні завдань до виконання, що є основною проблемою в сучасних хмарних ЦОД.

Відомим і широко використовуваним методом покращення ефективності використання ресурсів ЦОД є віртуалізація [1], що дає змогу консолідувати ВМ на одному ФС. Всі робочі навантаження в сучасних хмарних ЦОД обслуговуються за допомогою ВМ і контейнерів. У той же час, контейнери можуть виконуватись як в середовищі ВМ, так і на ФС безпосередньо. Таким чином, віртуалізація дозволяє спільне використання ресурсів ФС для обслуговування різних сервісів кінцевого споживача, забезпечуючи тим самим гарантії продуктивності та надійності. У сучасних хмарних ЦОД один застосунок (або сервіс), як правило, виконується однією або кількома примірниками гетерогенних ВМ [105] з метою оброблення робочого навантаження користувача.

Безперервні зміни робочого навантаження на сервіс користувача, перерозподіл графіків виконання ВМ, а також відмови ФС відбуваються непередбачуваним способом. Таким чином, ресурсною ємністю для обслуговування такого навантаження необхідно управляти динамічно і адаптивно, одночасно забезпечуючи гарантії дотримання рівня сервісу та продуктивності без порушення SLA. Провайдери хмарних сервісів застосовують різні методи для забезпечення достатньої кількості ФС, одночасно забезпечуючи

ефективне використання ресурсів, зменшуючи операційні витрати та енергоспоживання.

При управлінні необхідною потужністю ресурсів ЦОД для обслуговування робочого навантаження виникають дві проблеми. По-перше, проблема надмірного виділення ресурсів (over-provisioning) виникає, коли відповідно зарезервовані ресурси, визначені за допомогою пікового навантаження, перевищують вимоги в їх використанні протягом горизонту керування. Така відсутність еластичності призводить до втрати ресурсів у період роботи сервісу без пікових навантажень (більшість часу роботи сервісу) і, відповідно, неефективності використання електроенергії. По-друге, проблема недостатнього виділення ресурсів (under-provisioning) виникає, коли в хмарному ЦОД не має ресурсів, необхідних для обслуговування більшої кількості запитів існуючих та нових користувачів, що призводить до порушень SLA в частині затримки планування виконання ВМ/контейнера та, як наслідок, вірогідного зменшення кількості клієнтів.

Проблема управління продуктивністю ресурсів також повинна вирішуватись за допомогою ефективних методів консолідації ВМ. Щоб досягти виділення необхідної ємності ресурсів, необхідної для достатньої продуктивності з метою обслуговування запитів користувачів протягом горизонту керування, провайдери хмарних послуг повинні застосовувати адаптивні методи та алгоритми керування консолідацією ВМ. Управління консолідацією ВМ являє собою адаптацію до динамічних робочих навантажень через використання технології безперебійної міграції (live migration) ВМ. Ця технологія дозволяє ВМ переміщатись з недовантажених ФС, щоб мінімізувати кількість активних ФС, необхідних для обслуговування поточного навантаження користувача. Таким чином, вивільнені від навантаження ФС можуть бути переключені в "режим сну", або в режим мінімального споживання енергії з метою зменшення споживання енергії в ЦОД. У разі, коли зростає попит на відповідні ресурси, необхідну кількість ФС швидко можна повернути в робочий режим. У цьому підході існують два основних критерії: мінімізувати порушення SLA та мінімізувати споживання енергії в ЦОД.

У дисертаційній роботі запропоновано метод інтегрованого управління ресурсами ЦОД в рамках схеми реалізації стратегій S_3 і S_4 (табл. 2.1). Він полягає в управлінні необхідними ресурсами на базі динамічної моделі станів ЦОД, враховуючи споживання електроенергії та порушення SLA. Підхід до вирішення проблеми динамічного управління виділенням ресурсів заснований на моделі станів ЦОД з метою мінімізації споживання енергії ФС хмарного ЦОД при допустимій затримці розгортання ВМ в умовах нестачі або надлишку ресурсів і трендом на зменшення навантаження. Сформульовано проблему оптимізації, яка розглядає декілька типів ресурсів та гетерогенні ФС. При аналізі та моделюванні використані дані з журналів використання кластера Google [106]. Результати показують, що використання ІУР підходу дозволяє досягти енергозбереження при мінімізації порушень SLA. У дисертаційній роботі запропонована динамічна модель хмарного ЦОД, основна схема підходу ІУР; вхідні та вихідні сигнали контуру управління, модель споживання електроенергії з урахуванням міграції ВМ та гетерогенності ресурсів ФС, модель порушення SLA з урахуванням штрафів за перевантаження ФС та штрафів за затримку розгортання ВМ.

У дисертаційній роботі запропоновані основна схема реалізації управління методом ІУР; метод управління продуктивністю ЦОД, вхідні та вихідні сигнали контуру управління, модель споживання електроенергії з урахуванням міграції ВМ та гетерогенності ресурсів ФС, модель порушення SLA з урахуванням штрафів за перевантаження ФС та штрафів за затримку розгортання ВМ.

Постановка задачі

Одним із важливих етапів розроблення систем управління ресурсами ЦОД є моделювання процесів з урахуванням всіх показників, що чітко відображають стан системи у часі, використовуються в реальних умовах ЦОД і суттєво впливають на якість управління.

З цією метою необхідно розробити модель динаміки хмарного ЦОД, яка враховує різні типи ФС, різні типи ВМ і різні типи ресурсів в гетерогенному середовищі ЦОД. Крім того, модель повинна враховувати актуальний стан всіх об’єктів ЦОД, що приймають участь у процесі управління ресурсами; кількість ВМ, що працюють, створюються, мігрують і завершують роботу; кількість ФС,

що знаходяться в активному режимі і в режимі очікування (сну); кількість ВМ кожного типу, що потенційно можуть бути створені в ЦОД з урахуванням наявної ємності. Таким чином, модель динаміки хмарного ЦОД повинна використовуватися як базова для інших схем реалізації стратегій управління, розроблених у розділі 5.

3.5. Модель динаміки хмарного центру оброблення даних: стани та управляючі впливи

У працях [283, 284, 322] представлені моделі динаміки і методи управління ресурсами ЦОД, що реалізують інтегроване управління. Динамічне моделювання хмарного ЦОД виконується в дискретні моменти *t*, а рішення з управління ресурсами приймається на початку кожного інтервалу управління. Для опису моделі вводяться змінні, що описують стан системи, змінні для визначення дій керування та вхідні змінні.

В якості вхідних даних використовуються завдання, що надходять на вхід ЦОД, кожне з яких виконується або обслуговується в окремій ВМ. Для кожного завдання потрібні ресурси, такі як частка процесорного часу, об’єм пам’яті, пропускна здатність мережі та пропускна здатність системи збереження даних. Час виконання кожного завдання в загальному випадку не визначено. Це пояснюється тим, що в хмарних ЦОД час виконання певних типів завдань, як правило, невідомий системі управління. Завдання (або ВМ) працює доки не закінчиться завдання або клієнт не завершить його. Потреби ресурсів залежать від характеристик служб і робочих місць, що працюють всередині ВМ.

Змінні, що описують стан ЦОД

Визначимо такі змінні стану для динамічної моделі ЦОД в інтервалі між моментами t і $t+1$. Стан *i*-го ФС позначимо як $z_i(t) \in \{0,1\}$.

Використання *k*-го ресурсу *j*-ю ВМ на *i*-му ФС позначимо змінною $r_{ij}^k(t) \in [0,1]$ $i = \overline{1,M}$, $j = \overline{1,N}$, $k \in K$, де K – це набір типів ресурсів, таких як процесорна ємність, об’єм пам'яті, продуктивність вводу/виводу диска тощо. ВМ перестає існувати, якщо для кожного k $r_{ij}^k(t) = 0$. Значення $r_{ij}^k(t)$ нормалізується

до найбільшої *k*-ї ресурсної ємності ФС. Використання *k*-го ресурсу на *i*-му ФС позначається як $R_i^k(t) \in [0,1]$ $i = \overline{1,M}$, $k \in K$.

Позначимо як $a_{ij}^h(t) \in \{0,1\}$, $h = \overline{1,H}$, цілочисельну змінну, яка вказує, чи працює *j*-а ВМ типу *h* на *i*-му ФС ($a_{ij}^h(t) = 1$) або не використовується ($a_{ij}^h(t) = 0$). *H* позначає число типів ВМ.

Позначимо змінною $g = \overline{1,G}$ тип ФС, де *G* – це кількість типів ФС. У моделі необхідно враховувати гетерогенне середовище ЦОД та його вплив на процес прийняття рішень при управлінні ресурсами та ВМ. Тип ФС не змінюється у процесі управління. Тип *j*-ї ВМ, що працює на *i*-му ФС, позначається змінною $h_{ij}(t)$.

Об'єм ресурсу *k*, необхідний (замовлений при створенні) для *j*-ї ВМ, позначається змінною $c_j^k(t) \in [0,1]$ і нормалізований до найбільшої *k*-ї ресурсної спроможності ФС. Використання ресурсу *k j*-ю ВМ не повинно перевищувати замовлений об'єм, $r_j^k(t) \le c_j^k(t)$, крім випадків, коли змінено тип ВМ.

Об'єм наявного ресурсу *k i*-го ФС позначається змінною $C_i^k \in [0,1]$ і нормалізований до найбільшого об'єму ресурсу *k* на найбільш потужному ФС. Значення використання ресурсу та його об'єму нормалізуються таким чином, що найбільш потужний ФС має *k*-ту ресурсну ємність, що дорівнює 1.

Змінні управляючих впливів

Визначимо змінні, що позначають управляючі впливи динамічної моделі ЦОД протягом інтервалу між моментами *t* і $t+1$.

Позначимо як $u_{ij}(t) \in \{0,1\}$ цілочисельну змінну, яка вказує на міграцію *j*-ї ВМ з *i*-го ФС. Міграція відбувається тоді, коли $u_{ij}(t) = 1$. Позначимо змінною $x_{ij}(t) \in \{0,...,M\}$ індекс ФС, до якого *j*-та ВМ мігрує з *i*-го ФС. Якщо *j*-та ВМ не мігрує, тоді $x_{ij}(t) = i$. Позначимо змінною $y_i(t) \in \{0,...,N\}$ індекс ВМ, яка буде мігрувати з *i*-го ФС до ФС з індексом $x_{ij}(t)$. Якщо *j*-та ВМ не мігрує, тоді $y_i(t) = j$. Позначимо як $v_j(t) \in \{0,1\}$ цілочисельну змінну, яка вказує на сигнал управління

зміною типу *j*-ї ВМ. Якщо потрібно змінити тип *j*-ї ВМ, тоді $v_j(t)=1$, інакше $v_j(t)=0$.

Позначимо як $S_i(t)\in\{-1,0,1\}$ цілочисельну змінну, яка вказує на зміну стану *i*-го ФС. Якщо $S_i(t)=-1$, тоді *i*-й ФС повинен бути вимкнений, якщо $S_i(t)=1$, тоді *i*-й ФС повинен бути включений, якщо $S_i(t)=0$, тоді стан *i*-го ФС не змінюється. Позначимо як $s_{ij}(t)\in\{-1,0,1\}$ цілочисельну змінну, яка вказує на зміну стану *j*-ї ВМ. Якщо $s_{ij}(t)=-1$, тоді *j*-та ВМ повинна бути вимкнена, якщо $s_{ij}(t)=1$, тоді *j*-та ВМ повинна бути розгорнута на *i*-му ФС, якщо $s_{ij}(t)=0$, тоді стан *j*-ї ВМ не змінюється.

Рівняння стану, що відображають динаміку системи

Рівняння (3.5)-(3.9) описують динаміку системи, що моделюється. Наступний стан (динаміка) *i*-го ФС може бути виражений у такий спосіб (3.5):

$$z_i(t+1)=z_i(t)+S_i(t) \tag{3.5}$$

Динаміка роботи *j*-ї ВМ на *i*-му ФС може бути виражена у такий спосіб (3.6):

$$\begin{aligned} &a_{ij}(t+1)=(a_{ij}(t)+s_{ij}(t))(1-u_{ij}(t))+a_{x_{ij}(t)y_{ij}(t)}(t)u_{ij}(t)\\ &u_{ij}(t)=0\,|\,a_{ij}(t)=0. \end{aligned} \tag{3.6}$$

Динаміка використання *k*-го ресурсу *j*-ю ВМ на *i*-му ФС можна виразити у такий спосіб (3.7):

$$r_{ij}^k(t+1)=(a_{ij}(t)+s_{ij}(t))(r_{new}^k(t)s_{ij}(t)+r_{ij}^k(t))(1-u_{ij}(t))+r_{x_{ij}(t)y_{ij}(t)}^k(t)u_{ij}(t). \tag{3.7}$$

Зміна типу *j*-ї ВМ може бути виражена у такий спосіб (3.8):

$$\begin{aligned} &h_j(t+1)=\Big((a_{ij}(t)+s_{ij}(t))\Big(h_{ij}(t)\big(1-v_j(t)\big)+h_{new}v_j(t)\Big)\Big)\times\\ &\big(1-u_{ij}(t)\big)+h_{x_{ij}(t)y_{ij}(t)}(t)u_{ij}(t). \end{aligned} \tag{3.8}$$

Використання ресурсів *i*-го ФС для кожного ресурсу *k* повинно бути обмеженим. Таким чином, умова (3.9) повинна бути виконана.

$$\sum_{j=1}^{N} r_{ij}^k(t)\le C_i^k. \tag{3.9}$$

Позначимо змінною $W_{PM}(t)$ невідомий процес, що повільно змінюється. Цей процес впливає на кількість ФС, що обслуговують робоче навантаження.

Змінні та рівняння для обліку потужності

Число ФС, що працюють на поточний момент *t*, визначається як $M_{PM}(t) = \sum_{i=1}^{M} z_i(t)$, $M_{PM}(t) \in \mathbb{N}$, $M_{PM}(t) < M$. Чим нижче значення $M_{PM}(t)$ тим краща якість консолідації ВМ.

Число ВМ, що працюють на поточний момент *t*, визначається як $N_{VM}(t) = \sum_{i=1}^{M}\sum_{j=1}^{N} a_{ij}(t) z_i(t)$, $N_{VM}(t)$, $N_{VM}(t) \in \mathbb{N}$, $N_{VM}(t) < N$.

Число ВМ, які мігрують на поточний момент *t*, визначаються як $N_{mig}(t) = \sum_{i=1}^{M}\sum_{j=1}^{N} u_{ij}(t) z_i(t)$, $N_{mig}(t)$, $N_{mig}(t) \in \mathbb{N}$, $N_{mig}(t) < N$.

Число ВМ, що працюють на поточний момент *t* на *i*-му ФС, визначається як $N_i(t) = \sum_{j=1}^{N} a_{ij}(t)$.

Щоб оцінити кількість ВМ N' кожного типу $h \in H$, які теоретично можуть бути створені в ЦОД, пропонується використовувати таку функцію:

$$N' = \max_{h \in H}\left\{ \max_{k \in K}\left\{ \frac{\sum_{i=1}^{M} C_i^k}{c^k} \right\} \right\}. \square \qquad (3.10)$$

3.6. Моделі інтегрованого управління IT-інфраструктурою з консолідацією віртуальних машин

У дисертаційній роботі пропонуються нові стратегії управління потужністю ресурсів ЦОД та розподілом ВМ (включаючи перерозподіл) з урахуванням обмежень на енергоспоживання та порушення умов SLA. Структура об’єкта управління і базова схема реалізації стратегій S_3, S_4 представлено на рис. 2.6. Для управління кількістю активних ФС, для вибору нового місця розташування ВМ та для керування міграцією ВМ слід враховувати такі вимоги: (1) мінімізувати кількість порушень SLA, що виникають у зв'язку з

перевантаженням як мінімум одного з ресурсів ФС; (2) мінімізувати кількість порушень SLA, що виникають у зв'язку із затримкою розгортання нової ВМ; (3) мінімізувати споживання енергії в ЦОД з урахуванням гетерогенного та нестаціонарного стохастичного середовища IaaS. Для досягнення таких цілей запропоновано метод ІУР на основі моделі динаміки ЦОД.

У рамках запропонованого методу ІУР вирішуються такі задачі управління: консолідація ВМ, розміщення нової ВМ та задача планування ресурсної місткості. Запропонований метод базується на моделях енергоспоживання, порушення умов SLA та управління потужністю ЦОД, що використовують модель динаміки ЦОД.

3.6.1. Модель споживання електроенергії

Скорочення енергоспоживання центрів оброблення даних є актуальною проблемою на сьогодні. Енергоспоживання ЦОД складається зі споживання електроенергії для обслуговування навантаження клієнтів та енергоспоживання фізичної інфраструктури. Таким чином, важливо підвищити енергоефективність хмарних ЦОД через розроблення і впровадження моделей та алгоритмів систем управління та апаратного забезпечення, що безпосередньо обслуговують IT-навантаження. У роботі розглядається і вирішується задача моделювання енергоспоживання ФС у хмарному ЦОД. При цьому, задачі управління енергоспоживанням мережевого обладнання та пристроїв зберігання даних в запропонованій моделі не розглядаються і являють собою окремі підзадачі загальної задачі управління енергоспоживанням ЦОД.

Споживання електроенергії ФС визначається його компонентами, такими як процесор, чіпсет, накопичувач дисків, пам'ять, джерела живлення та системи охолодження [117]. На практиці, у гетерогенному середовищі неможливо окремо виміряти і враховувати при управлінні енергоспоживання кожного компонента. У [118] автори показали, що споживання електроенергії ФС може бути описане лінійною залежністю між електроспоживанням та використанням процесора. Інша проблема, розглянута в [118], це вузький динамічний діапазон потужності серверів, коли сервер, що працює в режимі очікування, витрачає близько 70% його максимальної потужності.

Слід також враховувати, що варіація інтенсивності робочого навантаження на кожну ВМ, розміщену на ФС, не впливає на споживання електроенергії центральним процесором ФС [109]. Електроспоживання ФС змінюється тільки при зміні кількості ВМ, що виконуються на ньому. Проте, неможливо визначити коефіцієнти, щоб врахувати внесок кожної ВМ в електроспоживання ФС, що їх містить. Таким чином, у дисертаційній роботі пропонується враховувати використання ресурсу $R_i^k(t)$ для кожного ресурсу k i-го ФС, оскільки $R_i^k(t)$ також вносить вклад у загальне споживання електроенергії ФС.

Позначимо змінною $e_{idle}^g \in \mathbb{R}^+$ споживання електроенергії ФС типу g коли він працює в режимі очікування. Гетерогенність ЦОД пропонується врахувати різними моделями споживання електроенергії, беручи до уваги коефіцієнт енергоефективності ФС типу g, що позначений як ρ^{gk}, для ресурсу типу k. Споживання електроенергії i-го ФС типу g, позначене змінною $e_i^g(t) \in \mathbb{R}^+$, може бути визначене у такий спосіб:

$$e_i^g(t) = e_{idle}^g + \sum_{k \in R} R_i^k \rho^{gk}. \tag{3.11}$$

Сумарне споживання електроенергії всіх працюючих ФС може бути визначене таким чином:

$$E_{PM}(t) = \sum_{i=1}^{M} z_i(t) e_i^g(t), \tag{3.12}$$

де $z_i(t) \in \{0,1\}$ – стан i-го ФС.

Інший важливий показник, який слід враховувати при моделюванні електроспоживання в хмарних ЦОД, це "жива" (без зупинки сервісу) міграція ВМ. За різними оцінками і залежно від кількості працюючих ФС, сотні ВМ в ЦОД можуть брати участь у процесі консолідації. Залежно від архітектури зберігання даних (зі спільним сховищем, або без спільного сховища) транспортні витрати на міграцію будуть різними. Якщо ЦОД використовує архітектуру без спільного сховища даних, витрати на передачу жорсткого диска ВМ додаються до витрат на транспортування пам'яті, що належить ВМ. У запропонованій моделі використовується архітектура зі спільним сховищем. Слід також

зазначити, що режим живої міграції не викликає суттєвого зменшення продуктивності роботи самої ВМ в порівнянні з продуктивністю роботи ВМ у звичайному режимі [5], але викликає навантаження на ФС, що беруть участь у процесі міграції ВМ.

Крім того, є обмеження гіпервізора щодо кількості одночасних міграцій з/на ФС, на завантаження підсистеми зберігання та мережевих адаптерів. Ці обмеження також повинні бути враховані. Встановлення обмежень кількості одночасних міграцій ВМ у реальному ЦОД виконується окремо для ФС та для його мережевого підключення. Міграція ВМ також є процесом з інтенсивним вводом/виводом, і енергія також споживається передавальними та приймальними пристроями на самих ФС, що беруть участь в управлінні процесом міграції, та мережевим обладнанням ЦОД [119].

Позначимо змінною $e_j^h(t)$ споживання електроенергії при міграції ВМ типу *h*. Тоді загальне електроспоживання при міграціях ВМ у поточний момент *t* можна визначити таким чином:

$$E_{mVM}(t) = \sum_{i=1}^{M}\sum_{j=1}^{N} u_{ij}(t) e_j^h(t), \tag{3.13}$$

де $u_{ij}(t) \in \{0,1\}$ – цілочисельна змінна, яка вказує на міграцію j-ї ВМ з i-го ФС. Міграція відбувається тоді, коли $u_{ij}(t) = 1$.

Споживання електроенергії центром оброблення даних у момент *t*, позначене змінною $E(t)$, може бути визначено таким чином:

$$E(t) = E_{PM}(t) + E_{mVM}(t). \tag{3.14}$$

3.6.2. Модель порушень умов SLA

Продуктивність ФС знижується коли один або більше його ресурсів використовуються на 100% [120], тому ВМ, що працюють на цьому ФС, в цьому випадку не забезпечені необхідними ресурсами для досягнення узгодженого рівня продуктивності. У реальних умовах обмеження використання деяких ресурсів може бути жорсткішим, від 85% до 95%, щоб не допускати вузьких місць [125]. Причиною цього є відсутність ресурсного резерву ФС для

обслуговування майбутніх потреб у ресурсах, що у решті-решт призводить до порушення SLA. Серед усіх ресурсів, що споживаються ВМ, є три найбільш важливих, які впливають на продуктивність ВМ, коли ФС завантажений на 100% навіть по одному з ресурсів деякий час.

По-перше, це використання процесора та довжина черги процесора. Продуктивність, яку надає процесор, знижується, коли використання становить 100%, а довжина черги процесора перевищує мінімально допустимий рівень. Іншими двома ресурсами є введення/вивід на пристрої збереження даних та на мережеві підключення. Ці ресурси можуть бути обмежені для кожної ВМ за допомогою гіпервізора, щоб уникнути перевантаження ФС. Але через деякий час ці два ресурси також можуть бути вичерпані через зміни кількості вхідних або вихідних запитів до підсистем збереження даних та до мережевих адаптерів з боку ВМ або системного програмного забезпечення ФС [329].

Пропонується визначити штрафи за порушення SLA, $P^{SLA}(t)$, у вигляді суми штрафів за перевантаження ФС, $P^{o}(t)$, та штрафів за затримки розгортання нової ВМ, $P^{d}(t)$, таким чином:

$$P^{SLA}(t) = p_1 P^{o}(t) + p_2 P^{d}(t), \tag{3.15}$$

де p_1 – узгоджена з користувачем вага одиниці штрафу за порушення SLA на одній ВМ, $p_1 > 0$; p_1 – узгоджена з користувачем вага одиниці штрафу за затримку розгортання ВМ, $p_2 > 0$.

Розглянемо розрахунок штрафів за перевантаження ФС по одному з ресурсів. Припустимо, що ФС може перебувати в нормальному стані (ресурси використовуються менше, ніж визначено порогом) або в перевантаженому стані. Перевантажений стан ФС настає тоді, коли показник використання одного з ресурсів досягає максимального, заздалегідь визначеного значення (порогу), близького до наявного обсягу цього ресурсу на ФС, протягом деякого періоду часу. Вважатимемо, що i-й ФС перевантажений, якщо $\exists k \in K : \sum_{j=1}^{N} r_{ij}^{k}(t) + r_{i}^{k}(t) = C_i^k$, де $r_{ij}^{k}(t) \in [0,1]$ $i = \overline{1,M}$, $j = \overline{1,N}$, $k \in K$ – використання k-го ресурсу j-ю ВМ на i-му ФС, $r_{i}^{k}(t)$ – це показник

використання k-го ресурсу самим ФС у момент t, K – це набір типів ресурсів, таких як процесорна ємність, об’єм пам'яті, продуктивність вводу/виводу диска тощо. Значення штрафу $P^o(t)$ за перевантажений стан ФС визначимо для кожного ресурсу $k \in K$ таким чином:

$$P^o(t) = N_{VM}^v(t), \tag{3.16}$$

де $N_{VM}^v(t)$ – кількість ВМ, на яких впливає перевантаження відповідного ФС. $N_{VM}^v(t)$ визначається таким чином:

$$N_{VM}^v(t) = \sum_{i=1}^{M} z_i(t)\psi(i), \tag{3.17}$$

якщо ресурс $k \in K$ існує такий, що $\psi(i)$ визначається як

$$\psi(i) = \begin{cases} 0, & \dfrac{C_i^k}{\sum_{j=1}^{N} r_{ij}^k(t) + r_i^k(t)} < 1, \\ \sum_{j=1}^{N} a_{ij}(t), & \textit{інакше} \end{cases}. \tag{3.18}$$

Розглянемо розрахунок штрафів за затримку розгортання нової ВМ $P^d(t)$. Припустимо, що всі ВМ повинні бути розгорнуті в інтервалі між моментами t і $t+1$ відповідно до SLA. Ця вимога стосується ВМ, які створюються для забезпечення певної продуктивності роботи сервісу або балансування навантаження на сервіс.

Наприклад, якщо SLA визначає, що час розгортання повинен бути не більше, ніж часовий інтервал τ між t і $t+1$, це означає, що в середньому, час, потрібний для розгортання екземпляру ВМ, складає τ. Постачальник хмарних послуг зацікавлений у мінімізації кількості задіяних (працюючих) ФС при достатній наявній кількості ресурсів. Таку ціль можна досягнути за рахунок консолідації ВМ та наявності групи вільних ФС, щоб бути готовим до прийому нових ВМ у міру необхідності. Кількість ФС, готових для прийняття нових ВМ, може бути оцінена за допомогою моделі управління ємністю.

Кількість ВМ на наступному кроці $t+1$ управління ресурсами ЦОД визначається таким чином:

$$N_{VM}(t+1) = N_{VM}(t) + N_{VM}^{on}(t) - N_{VM}^{off}(t), \quad (3.19)$$

де $N_{VM}^{on}(t)$ – кількість ВМ, визначених на розгортання, $N_{VM}^{off}(t)$ – кількість ВМ, визначених на вимикання. Значення $N_{VM}^{off}(t)$ коректується кожного разу на наступному кроці управління ресурсами ЦОД, щоб врахувати аномальне вимкнення ВМ.

Кількість ВМ, запланованих до розгортання, визначається таким чином:

$$N_{VM}^{on}(t) = \sum_{j=1}^{N} \gamma_1(s) s_{ij}(t), \quad (3.20)$$

де $\gamma_1(s)$ – це функція, визначена так, що $\gamma_1(s) = 1$, якщо $s = 1$, та $\gamma_1(s) = 0$ в іншому випадку; $s_{ij}(t) \in \{-1, 0, 1\}$ – цілочисельна змінна, яка вказує на зміну стану j-ї ВМ, якщо $s_{ij}(t) = -1$, тоді j-та ВМ повинна бути вимкнена, якщо $s_{ij}(t) = 1$, тоді j-та ВМ повинна бути розгорнута на i-му ФС, якщо $s_{ij}(t) = 0$, тоді стан j-ї ВМ не змінюється.

Кількість ВМ, запланованих для завершення роботи, визначається таким чином:

$$N_{VM}^{off}(t) = \sum_{j=1}^{N} \left| \gamma_2(s) s_{ij}(t) \right|, \quad (3.21)$$

де $\gamma_2(s)$ – це функція, визначена так, що $\gamma_2(s) = -1$, якщо $s = -1$, та $\gamma_2(s) = 0$ в іншому випадку.

За кожний додатковий час τ, витрачений на розгортання ВМ, постачальник хмарних послуг сплачує штраф P. Позначимо змінною $w(t)$ число ВМ, які не були розгорнуті на попередньому кроці від $t-1$ до t. Таким чином, штраф $P^d(t)$ за затримку розгортання ВМ за визначений часовий період $\Upsilon = \{1, 2, ..., T\}$ визначається таким чином:

$$P^d(t) = \sum_{t=1}^{T} w(t). \quad (3.22)$$

3.6.3. Модель планування потужності

Одним з найважливіших завдань управління ресурсами ЦОД є підтримка ресурсної потужності на необхідному рівні, достатньому для розгортання нових ВМ у відповідний час. У разі збільшення кількості запитів від користувачів до сервісу, ще одна (або декілька) ВМ повинні бути розгорнуті для дотримання SLA, визначеної в термінах середнього часу відгуку сервісу. Відсутність вільних ресурсів може спричинити штраф за недотримання SLA та суттєву затримку розгортання ВМ, тобто час очікування виконання завдання може збільшитися на час від однієї до трьох хвилин поки буде ввімкнений ще один (або декілька) ФС.

Пропонується взяти до уваги два основних процеси, які відбуваються одночасно і безпосередньо впливають на поточну ресурсну ємність ЦОД, а саме врахувати: $N_{VM}^{on}(t)$ – кількість нових ВМ для розгортання в момент *t*, та $N_{VM}^{off}(t)$ – кількість ВМ, що вимкнені в момент *t*. Також, можна отримати реальні дані в момент *t* від ФС щодо ВМ, які зникли унаслідок помилок і збоїв. Тут *t* розглядається як довільний момент часу, коли визначається стан моделі для прийняття наступного рішення.

У дисертаційній роботі пропонується дві нові метрики. Перша метрика – миттєвий коефіцієнт життєздатності ВМ, позначений змінною V_{VM}^{inst}, для обліку динаміки роботи ВМ в хмарному ЦОД в момент *t* (3.23). Миттєвий коефіцієнт життєздатності пропонується використовувати для прийняття рішень на найближчий крок управління ресурсами ЦОД і може бути обчислений таким чином:

$$V_{VM}^{inst} = \frac{N_{VM}^{on}(t)}{N_{VM}^{off}(t)}.. \tag{3.23}$$

Коефіцієнт V_{VM}^{inst} пропонується використовувати для вибору стратегії управління та для управління міграціями. Якщо $V_{VM}^{inst} < 1$ для короткострокового горизонту управління (1-3 кроки), то стратегія, яка в тому числі управляє кількістю працюючих ФС, що обслуговують ВМ, може бути скоригована для більш агресивного виконання консолідації ВМ. Якщо $V_{VM}^{inst} > 1$ для

короткострокового горизонту управління, то стратегія, в рамках якої відбувається управління кількістю працюючих ФС, повинна зменшити кількість міграцій або взагалі припинити процес консолідації ВМ.

Оскільки миттєве значення не дає інформацію про минулі зміни в кількості ВМ та одного інтервалу управління (часу між моментами t і t+1) не достатньо для включення ФС з додаванням його в пул доступних ресурсів, то пропонується використовувати другу метрику – середній коефіцієнт життєздатності ВМ, позначений змінною V_{VM}^{mid}, для обліку динаміки змін кількості ВМ в хмарному ЦОД. V_{VM}^{mid} пропонується обчислювати одним з варіантів зваженого ковзного середнього. Такий вибір обумовлений відсутністю великих відносних коливань значень $N_{VM}^{on}(t)$ і $N_{VM}^{off}(t)$. Інтервал усереднення при цьому повинен бути узгоджений з найбільшим часом включення ФС в ЦОД (або кластері).

Якщо $V_{VM}^{mid} < 1$ для середньострокового горизонту управління (більше 4-х кроків управління), то стратегія, яка визначена для управління кількістю працюючих ФС, що обслуговують ВМ, полягає у більш агресивному виконанні консолідації ВМ з урахуванням обмежень на міграції. Якщо $V_{VM}^{mid} > 1$ для середньострокового горизонту управління, то стратегія, яка визначена для управління кількістю працюючих ФС, повинна припинити процес консолідації ВМ і включити додаткові ФС певної ємності з додаванням їх в пул доступних ресурсів використовуючи запропонований метод управління потужністю.

3.7. Модель і метод оптимального перерозподілу ресурсів IT-інфраструктури

Рішення з управління ресурсами включають в себе рішення щодо консолідації ВМ, рішення про міграцію ВМ, рішення про управління станом ФС, а також рішення про розміщення нової ВМ. Ці рішення можуть прийматися одночасно в гетерогенних середовищах IaaS. Для моделювання динаміки системи час поділено на інтервали рівної тривалості, поточний крок управління позначений як t. Рішення з управління відбуваються на початку кожного інтервалу.

На початку кожного інтервалу управління *t*, кількість ФС, які будуть потрібні на наступний момент *t*+1, визначається з використанням моделі динаміки ЦОД, моделі планування потужності, та методу управління потужністю. Мета планування потужності – запланувати своєчасне включення ФС. Інформація про стан ЦОД в момент *t* спостерігається за допомогою менеджера ФС (МФС) та менеджера ВМ (МВМ) (рис. 2.3), в тому числі значення $M_{PM}(t)$, $N_{VM}^{on}(t)$ і $N_{VM}^{off}(t)$. Використовуючи метод управління потужністю ЦОД, можна визначити кількість ФС, необхідних на момент *t*+1, і визначити відповідні керуючі впливи.

Загальна мета ІУР ЦОД полягає в тому, щоб управляти кількістю ФС з урахуванням обмежень і згідно критерію енергозбереження та дотримання SLA. У запропонованій моделі, управління кількістю ФС виконується за допомогою контрольної змінної $S_{ij}(t)$, через визначення змінних розташування ВМ $x_{ij}(t)$ і $y_{ij}(t)$, та через визначення змінної міграції $u_{ij}(t)$ таким чином, щоб мінімізувати загальну суму штрафів з точки зору міграції ВМ та затримки планування виконання ВМ (3.15) при мінімізації споживання електроенергії (3.14). Задача ІУР зводиться до мінімізації на кожному кроці управління *t* функції:

$$J = \alpha \left| P^{SLA}(t) \right|^2 + \beta \left| E(t) \right|^2, \tag{3.24}$$

за умов (3.1) – (3.9),

де α і β – це ваги, визначені користувачем, які позначають відносну важливість складових критерію.

Задача ІУР ЦОД (3.24) є задачею нелінійного цілочисельного програмування. Її оптимальне рішення можна знайти за допомогою класичних методів оптимізації. Але, якщо взяти до уваги велику кількість змінних та обмежень, класичні методи оптимізації будуть вимагати значної кількості обчислень для оптимізації, що в режимі онлайн складно досягнути.

Для того, щоб застосувати ІУР ЦОД у режимі реального часу для великомасштабних ЦОД, пропонується вирішувати задачу оптимізації (3.24) приблизно, за рахунок використання оптимізації Монте-Карло [121]. Ідея застосування наближених методів полягає в тому, щоб випадково обирати

велику кількість зразків контрольних входів і обчислювати функцію вартості для кожного з них. Отримана функція вартості з найнижчим значенням є субоптимальним керуванням ресурсами ЦОД в момент *t*. Необхідно тільки підібрати відповідну кількість зразків для налагодження компромісу між часом розрахунку та точністю наближення.

Для оцінки ефективності підходу ІУР ЦОД та запропонованих евристик модель динаміки ЦОД реалізована за допомогою мови C#, а також імітовані різні сценарії на вхідних даних з трейсів (журналів статистики) використання кластера Google [106].

3.8. Структурні вимоги до реалізації інтегрованого управління ресурсами

В дисертації розроблений метод ІУР, призначений для розв'язання задач консолідації ВМ, розміщення нової ВМ та планування ресурсної місткості в якості схем реалізації стратегій S_3 та S_4. Структура реалізації централізованого управління хмарним ЦОД показана на рис. 3.2. ЦОД представлений набором ФС, що характеризуються певною конфігурацією апаратної та програмної підсистем та фіксованою ємністю ресурсів. На кожному ФС може бути розміщено декілька ВМ. ФС поєднані мережею ЦОД один з одним та централізованою системою зберігання даних (сховищем) [16].

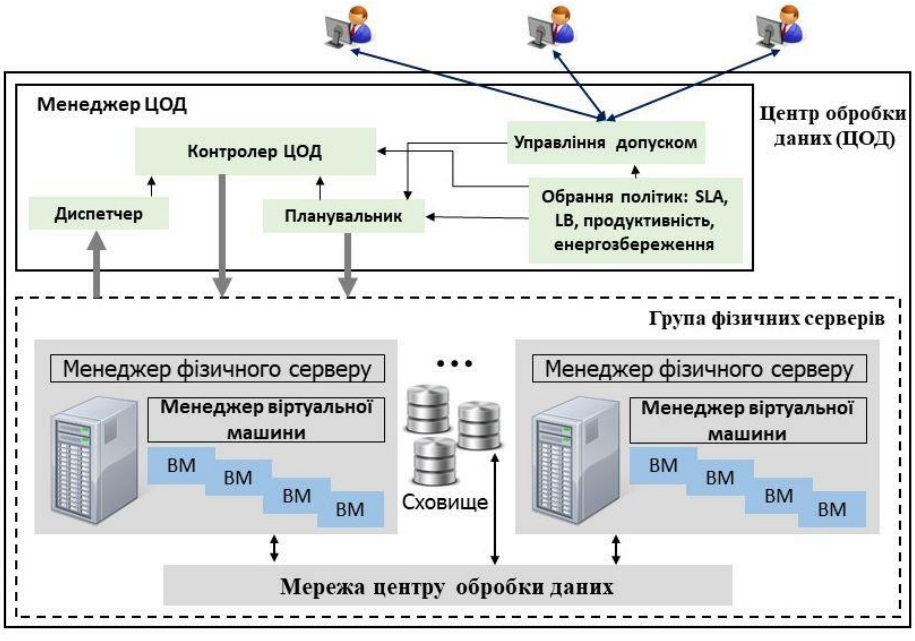


Рис. 3.2. Структура реалізації централізованого управління ІТ-інфраструктурою хмарного ЦОД

Модель ЦОД складається з $M \in \mathbb{N}$ ФС і $N \in \mathbb{N}$ ВМ. Для того, щоб враховувати динаміку процесів ЦОД, позначимо змінною N максимально допустиму кількість ВМ з мінімальними вимогами до ресурсів, тому N є функцією від M та вимог до ресурсів. Таким чином, кількість ФС і ВМ може змінюватися через несправності, відновлення та/або додавання нових ФС.

На практиці, в ЦОД, кількість ВМ завжди змінюється. Необхідно враховувати, що в динамічному середовищі певні ВМ, визначені для міграції, можуть перестати існувати під час управління. Тому, пропонується оцінити кількість ВМ N', які теоретично можна розмістити в ЦОД в момент часу t, використовуючи функцію оцінки, яка враховує конфігурацію ВМ з мінімальними вимогами до ресурсів та загальною ємністю ресурсу ЦОД.

У гетерогенному середовищі кожен ФС має типову конфігурацію ресурсів, що включає процесор, оперативну пам'ять, сховище та підключення до мережі, з метою надання їх визначеній кількості ВМ при розміщенні. На кожному ФС виконується МВМ, такий як Xen, ESXi або Hyper-V, а також виконується спеціальна ВМ, така як МФС для керування ВМ засобами МВМ. Використовуючи показники моніторингу, МФС отримує поточне використання ресурсів ФС та використання ресурсів кожною розміщеною на ньому ВМ. Рішення з управління ресурсами приймаються менеджером ЦОД (рис. 3.2), який виконує консолідацію (або міграцію) ВМ, управління станом ФС і ВМ, а також визначає розміщення нової створюваної ВМ. На рівні програмного забезпечення існує множина клієнтів, які потребують надання послуг або застосунків, що забезпечуються однією або декількома ВМ. Продуктивність кожної ВМ обмежена конкретними показниками, зазначеними в SLA (наприклад, час на обслуговування запиту, час роботи, кількість запитів у секунду та ін.).

3.9. Дворівнева модель системи управління гіперконвергентною IT-інфраструктурою

В дисертаційній роботі пропонується вирішувати завдання управління ресурсами ГІ за допомогою системи управління ГІ (СУГІ), розробленої в [282] показаної на рис. 3.3. СУГІ - це розвиток системи управління дворівневою IT-інфраструктурою, запропонованою в [249].

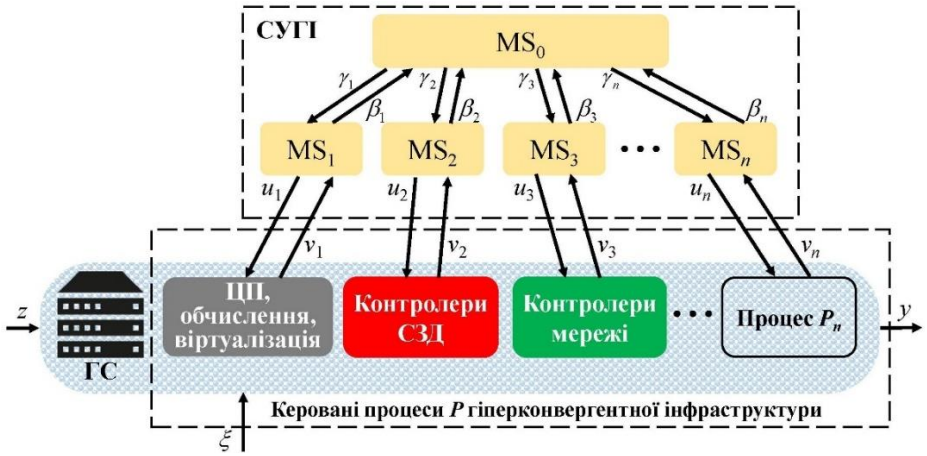


Рис. 3.3. Модель системи управління ГІ

Розташування підсистем управління (англ. management systems, MS) відображає ієрархічну структуру СУГІ. Модель складається з підсистеми управління вищого рівня (MS_0), а саме координатора, n підсистем управління другого рівня (MS_1,..., MS_n) і керованих процесів P, які існують в ГІ. Процеси P представлені процесами управління кожною підсистемою ГІ, такою як процес управління СЗД, процес управління мережею, процес управління віртуалізацією та інші.

Взаємодія підсистем управління вздовж вертикалі виконується у такий спосіб. Команди та управляючі впливи $\gamma_1,...,\gamma_n$, передані з MS_0 на MS_1,...,MS_n, є координуючими впливами. Команди і управляючі впливи $(u_1,...,u_n)$ від MS_1,...,MS_n до процесів P є керуючими впливами. Сигнали зворотного зв'язку $v_1,...,v_n$ передаються від процесів P до MS_1,...,MS_n. Сигнали зворотного зв'язку $\beta_1,...,\beta_n$. передаються від підсистем управління до координатора. Опис дворівневої СУГІ може бути реалізований за допомогою термінальних змінних, таких як входи і виходи.

Керовані підсистеми P піддаються впливу сигналів управління u з MS_1,...,MS_n, де U - набір керуючих впливів. На керовані підсистеми P впливають також вхідні сигнали z, $z \in Z$, де Z - множина запитів користувачів і сигнали ξ, $\xi \in \Xi$, де Ξ позначає збурюючі впливи. Збурюючі впливи включають несправності в ГІ, функціональні збої в елементах і підсистемах ГІ, перевантажений стан підсистеми. Вихідні сигнали процесів P позначені як y, $y \in Y$, де Y - набір виходів підсистем, таких як дані моніторингу.

Процеси P можуть бути представлені у вигляді відображення на основі декартового добутку $P: U \times Z \times \Xi \to Y$. Набір U керуючих впливів, що впливають на процеси P, можна представити у вигляді декартового добутку n наборів $U = U_1 \times U_2 \times ... \times U_n$. [70]. У цьому випадку кожна підсистема управління $MS_1,...,MS_n$ має повноваження вибирати окремі компоненти з $u_1,\ldots,u_n$ щоб мати прямий вплив на процеси P.

Кожна i-та підсистема управління MS_i, $i = \overline{1,n}$ отримує два вхідних сигнали: координаційний вплив $\gamma_i \in \Gamma$ від MS_0 і сигнал зворотного зв'язку v_i у вигляді даних моніторингу. Вихід управління MS_i, це вплив u_i, обраний MS_i з набору U_i. Припустимо, що кожен MS_i реалізує відображення C_i у такий спосіб $C_i: \Gamma \times V_i \to U_i$, де V_i - набір даних моніторингу v_i, що надходять до СУГІ з підсистем ГІ, $v_i \in V_i$. Дані моніторингу V_i, $i = \overline{1,n}$ - це сигнали зворотного зв'язку всередині i-ї підсистеми ГІ.

Сигнали зворотного зв'язку v_i, що надходять на вхід MS_i, отримуються з підсистем ГІ унаслідок моніторингу. Природно, що ці дані функціонально залежать від керуючих сигналів u, входів z і збурюючих впливів. Ця залежність може бути представлена таким відображенням [70]:

$$f_i: U \times Z \times \Xi \times Y \to V_i. \tag{3.25}$$

Підсистема управління MS_0 є координатором, який генерує координуючі сигнали $\gamma_i \in \Gamma$ і сигнал з i-го виходу MS_0 йде тільки на вхід i-ої нижньої підсистеми управління MS_i. Координатор формує сигнал на основі аналізу даних, що надходять на його входи від MS_i, і представляють собою сигнали зворотного зв'язку про стан і функціонування відповідних підсистем ГІ. У цьому випадку можна припустити, що в координаторі відображення C_0 реалізується у такий спосіб

$$C_0: \mathrm{B} \to \Gamma, \tag{3.26}$$

де B - набір інформаційних сигналів β, що реалізують зворотний зв'язок, $\beta = (\beta_1,...,\beta_n)$ - сукупність сигналів зворотного зв'язку β_i, що надходять до координатора з підсистем MS_i.

Подібно до (3.25), сигнал зворотного зв'язку β, прийнятий в MS_0, несе інформацію про стан всіх підсистем, що лежать рівнем нижче, тому визначається відображенням $f_0 : \Gamma \times V \times U \to B$, де $V = V_1 \times ... \times V_n$. Таким чином, B є функцією координатних сигналів γ_i, сигналів зворотного зв'язку $v = (v_1,...,v_n)$, що надходять на MS_i, і керуючих впливів $u = (u_1,...,u_n)$.

На моделі, показаної на рис. 3.3, взаємодія між підсистемами управління $MS_1,...,MS_n$ явно не показана, а також не вказаний прямий вплив MS_0 на функціонування ГІ, який може мати місце в реальній ГС.

Відповідно до [70], координація ґрунтується на впливі на підсистеми управління MS_i, що змушує їх діяти узгоджено, піддаючи дії керованих підсистем єдиній політиці, спрямованій на досягнення глобальної мети ГІ, незважаючи на те, що ця мета може суперечити локальним цілям підсистем. MS_0 виконує координацію і долає суперечності між локальними цілями підсистем MS_i.

Успішність діяльності координатора при організації узгоджених дій MS_i оцінюється тим, наскільки успішно досягнута глобальна мета управління ГІ. Досягнення мети координатором можна розглядати як вирішення задачі, яку можна формалізувати як проблему прийняття рішень, що полягає у оцінці ефективності процесу координації. Оскільки ця задача визначається відносно всіх підсистем управління, включаючи процеси *P*, то вона визначається як глобальна задача [70].

Для дворівневих систем координація повинна бути забезпечена відносно задачі, розв'язуваної MS_0, і забезпечувати її вирішення у зв'язку з вирішенням глобальної задачі. Перше означає, що сигнали від MS_0 мають координаційний вплив на завдання, що вирішуються MS_i, а другий означає, що координатор здатний впливати на MS_i так, що їх комбінований вплив на процеси *P* буде спрямований на вирішення глобальної задачі.

Успішне функціонування СУГІ, що побудована на основі дворівневої моделі, може бути забезпечено, коли цілі підсистем узгоджуються між собою і узгоджуються з глобальною метою системи [70]. У дворівневій системі виділяються три типи цілей: глобальна мета, мета координатора MS_0 і цілі

керуючих підсистем MS_i. Необхідність узгодженості цілей випливає з таких особливостей. На процес *P* безпосередньо впливають лише MS_i, тому глобальна мета може бути досягнута лише непрямим шляхом через дії MS_i, які повинні бути узгоджені щодо глобальної мети, а також мети координатора.

Глобальною метою є підвищення ефективності надання хмарних послуг. Вона виходить за рамки безпосередньої діяльності дворівневої системи, показаної на рис. 3.3, і жодна з підсистем MS_i не орієнтована на досягнення глобальної мети або вирішення глобального завдання. Глобальна задача може бути вирішена тільки спільними діями всіх підсистем управління MS_i.

Глобальна мета. Враховуючи той факт, що ГІ розгортаються для підвищення ефективності роботи сервісів при обслуговуванні клієнтів, глобальна мета СУГІ може бути визначена як забезпечення максимальної якості надання ІТ-послуг з мінімальними витратами при забезпеченні показників SLA. Максимальна якість надання ІТ-послуг позначається як $\max \mathfrak{Q}$.

Максимальна якість ІТ-послуг, що надаються ГІ, буде досягнута, коли

$$\max \mathfrak{Q} \Leftrightarrow \max Q_j, \forall j = \overline{1,K} \Leftrightarrow \max q_{kj}, \forall j = \overline{1,K}, \forall k = \overline{1,M_j},$$

де Q_j, $j = \overline{1,K}$ - якість роботи *j*-го сервісу; q_{kj}, $k = \overline{1,M_j}$ - значення *k*-го показника якості *j*-го сервісу.

Для досягнення мети управління ІТ-інфраструктурою ГС в цілому необхідно постійно розширювати і масштабувати апаратні ресурси ІТ-інфраструктури, що неприйнятно з економічної точки зору. Це обумовлено наявністю тимчасових пікових навантажень З іншого боку, підвищення економічної ефективності надання ІТ-послуг вимагає зменшення загальної вартості володіння (ЗВВ, англ. TCO), тобто дій, спрямованих на досягнення $\min \mathfrak{C}$.. Підтримка якості надання ІТ-послуг на заданому рівні є головним завданням координатора.

Мета координатора. Метою координатора є підтримка якості ІТ-послуг $\mathfrak{Q}$ на узгодженому рівні з мінімальними витратами $\mathfrak{C}$ на залучені ресурси ІТ-інфраструктури. Мета координатора може бути формалізована у такий спосіб:

$$\mathfrak{Q} = \text{const}\big|_{\min \mathfrak{C}}. \qquad (3.27)$$

Вираз (3.27) означає, що координатор з усіх можливих впливів вибере ті, які потребують мінімальної вартості реалізації.

Вимога щодо підтримки узгодженого рівня надання ІТ-послуг охоплює всі послуги та індивідуальні показники якості послуг:

$$\Omega = \text{const} \Leftrightarrow Q_j = \text{const}, \forall j = \overline{1,K} \Leftrightarrow q_{kj} = \text{const}, \forall k = \overline{1,M_j}, \forall j = \overline{1,K}.$$

Основним способом підвищення якості роботи *j*-го сервісу є виділення додаткових ресурсів, що підтримують *j*-й сервіс на всіх рівнях. Якщо якість надання *j*-го сервісу перевищує задане значення, ресурси, що відповідно виділяються на всіх рівнях, треба зменшити, як того вимагає критерій $\min C$. У той же час, останній сервер, який надає *j*-у ІТ-послугу, не може бути вимкнений, незважаючи на те, що якість цієї послуги все ще вище, ніж потрібно, оскільки це призведе до повного припинення надання послуги. Але все ще є можливість мігрувати ВМ, в якій працює сервіс, на інший ФС. Таким чином, завжди буде деякий фіксований мінімум витрат, після чого подальше зниження витрат буде неможливим.

Локальні цілі кожної підсистеми. Метою локального управління є підтримка заданих значень параметрів функціонування ГІ з мінімальними витратами, тобто забезпечення $q_{kj} = \text{const}\big|_{\min C}, \forall k = \overline{1,M_j}, \forall j = \overline{1,K}$.

У моделі СУГІ, показаної на рис. 3.3, підсистеми управління MS_i, можуть мати свої власні цілі функціонування, тобто можуть вирішуватись, також, додаткові задачі, наприклад, з резервного копіювання, балансування навантаження та ін.

Висновки до розділу 3

1. Розроблено модель динаміки хмарного ЦОД, яка враховує різні типи ФС, різні типи ВМ і різні типи ресурсів в гетерогенному середовищі ЦОД. Розроблена модель дозволяє отримати актуальний стан всіх об’єктів ЦОД, що приймають участь у процесі управління ресурсами, в певний дискретний момент часу, а також визначити кількість працюючих ВМ; кількість мігруючих ВМ, кількість задіяних ФС, кількість ВМ кожного типу, що потенційно можуть бути

створені в ЦОД з урахуванням наявної ємності. У результаті аналізу журналів роботи кластера Google виявлено значне коливання попиту на кожний тип ресурсу в часі та затримка запуску ВМ для виконання завдання.

2. Розроблений метод ІУР ЦОД забезпечує необхідну кількість ФС для оброблення навантаження зі зменшенням електроспоживання та зменшенням кількості порушень угоди про рівень обслуговування клієнтів SLA. Метод ІУР ЦОД базується на моделі динаміки хмарного ЦОД, яка враховує гетерогенність ФС та ВМ, зміну їх станів і конфігурацій, можливість управління розміщенням нових ВМ та їх міграцією. Розроблена модель споживання електроенергії дозволяє мінімізувати споживання енергії усіма ФС одночасно підтримуючи прийнятну затримку розгортання нової ВМ. Модель обліку порушень SLA враховує штрафи за перевантаження ФС і штрафи за затримку розгортання нової ВМ. Розроблена модель планування потужності ЦОД дозволяє визначити кількість ФС, необхідних для забезпечення обслуговування прогнозованого навантаження у вигляді ВМ з певними вимогами до ресурсів.

Аналіз та моделювання з використанням трейсів роботи кластера Google [106], представлені у розділі 9, показують, що за допомогою підходу ІУР ЦОД постачальники хмарних сервісів можуть зменшити споживання електроенергії при мінімізації порушень SLA з точки зору кількості перевантажених ФС та затримки розгортання нової ВМ.

3. Розроблено архітектуру багаторівневої ієрархічної системи управління ІТ-інфраструктурою хмарного ЦОД. Декомпозиція ІТ-інфраструктури ЦОД на рівні є загальним підходом, що полягає у розкладання складної системи на рівні управління. ІТ-інфраструктура хмарного ЦОД розділена на три рівні (рівень інфраструктури, рівень платформи та прикладний рівень). Запропонована модель використовує набори різних параметрів, визначених в хмарному ЦОД, таких як: набір послуг, набір управлінських впливів на кожному шарі моделі, набір показників якості обслуговування і набір споживаних ресурсів. Запропоновано оцінювати ефективність управління послугами за якістю наданих послуг, за витратами на використання ресурсів ЦОД та за витратами за порушення SLA. На основі результатів декомпозиції пропонується підхід до розроблення багаторівневої ієрархічної системи управління. Кінцевою метою є

управління ІТ-інфраструктурою хмарного ЦОД за критерієм мінімізації витрат і мінімізації відхилень параметрів якості надання послуг від заданих на кожному рівні. Багаторівнева ієрархічна система управління дозволяє забезпечити ефективне надання послуг та застосунків на вищих рівнях, впливаючи на параметри продуктивності інфраструктурного рівня та впливаючи на конструктивні конфігурації на рівні платформи та прикладному рівні.

4. Розроблено дворівневу модель системи управління гіперконвергентною ІТ-інфраструктурою, на верхньому рівні якої функціонує координатор роботи підсистем, а на нижньому рівні функціонують програмно-визначені підсистеми керування сховищами, мережею та гіпервізорами. Оскільки гіперконвергентна система обслуговує різні навантаження різних ІТ-сервісів потрібно координувати роботу підсистем нижнього рівня з метою досягнення заданих показників якості в умовах обмежень, що накладаються на всю систему в цілому.

РОЗДІЛ 4. МОДЕЛІ І МЕТОДИ ПРОГНОЗУВАННЯ СПОЖИВАННЯ РЕСУРСІВ І НАВАНТАЖЕННЯ В ХМАРНИХ ЦОД

У розділі обґрунтовано необхідність прогнозування навантаження на ресурси ІТ-інфраструктури хмарного ЦОД, сформульовано задачу прогнозування навантаження на обчислювальні ресурси, проаналізовано існуючі моделі прогнозування, розроблено моделі і методи адаптивного прогнозування навантаження на ресурси хмарного ЦОД із застосуванням набору альтернативних методів і моделей прогнозування, змінного розміру навчальної вибірки та евристик комбінування прогнозних значень.

4.1. Прогнозування навантаження на ІТ-інфраструктуру

Оскільки для реалізації стратегій управління ресурсами ІТ-інфраструктури хмарного ЦОД необхідно використовувати прогнози навантаження і інших метрик, що використовуються при визначенні стратегії, необхідним є розроблення в дисертаційній роботі методів прогнозування, які адаптуються до поточних умов роботи хмарного ЦОД. Прогнозування, також, використовується для забезпечення роботи методів управління, розроблених в дисертаційній роботі.

В сучасних умовах конкуренції між провайдерами хмарних послуг все більш актуальним стає питання забезпечення заданої якості надання сервісів кінцевому користувачеві. До кожного сервісу висуваються вимоги високої доступності та продуктивності, що забезпечуються через масштабування, балансування навантаження, резервування та застосуванням ефективних методів управління ємністю фізичних та віртуальних ресурсів.

ІТ-інфраструктура, що забезпечує роботу віртуалізованих застосунків, обслуговує різні комбінації змінних навантажень. У цих умовах провайдер хмарних послуг повинен забезпечити виконання угоди щодо дотримання рівня сервісу SLA. Зазвичай, реактивний підхід до управління обчислювальними ресурсами не забезпечує належний рівень надання сервісу, що відбивається, в решті решт, на кількості клієнтів та репутації провайдера. Актуальним є

проактивний підхід до управління обчислювальними ресурсами хмарного ЦОД, який полягає у застосуванні методів прогнозування.

Одним із найважливіших механізмів, що забезпечує роботу хмарних обчислювальних систем є консолідація навантажень в рамках однієї ресурсної одиниці – ФС. Завдяки застосуванню віртуалізації [1] такі сутності, як ВМ та контейнери розміщуються на одному ФС з метою більш ефективного використання наявних фізичних ресурсів. Таким чином, якщо отримати прогнозоване значення споживання ресурсів з боку ВМ або контейнера з'являється можливість запобігти виникненню проблем недостатнього виділення ресурсів (under-provisioning) та надмірного виділення ресурсів (over-provisioning). У першому випадку виникає уповільнення роботи сервісів і порушення SLA, а в другому – збільшуються витрати електроенергії за рахунок збільшеної кількості увімкнених ФС.

Проблема прогнозування робочого навантаження ФС не є тривіальною, оскільки потреби ВМ в ресурсах можуть значно змінюватися під впливом числа клієнтських запитів і конкретних програм, які працюють всередині ВМ. Розробленню моделей і методів прогнозування навантаження присвячені роботи [286, 327].

Розглянути всі можливі комбінації розміщення ВМ з конкретними потребами в ресурсах не представляється можливим. Крім того, алгоритми прогнозування, як правило, повинні бути обрані відповідно до конкретних умов використання у процесі роботи серверної системи.

Оскільки умови роботи ФС залежать від багатьох випадкових факторів, як зовнішніх, так і внутрішніх, необхідно адаптуватися до різних умов при прогнозуванні потреби в ресурсах з використанням структурної адаптації. У цьому випадку структурна адаптація полягає в тому, щоб вибрати відповідний алгоритм прогнозування для поточних умов експлуатації ФС.

Для того, щоб виконати прогноз робочого навантаження ФС пропонується використовувати такі алгоритми прогнозування, які показали хороші результати при прогнозуванні використання ресурсів для різних робочих навантажень ВМ в різних умовах: авторегресійне інтегроване ковзне середнє [13], експоненційно зважене ковзне середнє [5] і розріджені періодичні авторегресії [2].

4.2. Задача прогнозування навантаження на обчислювальний ресурс

Для роботи ФС або ВМ необхідні такі ресурси: процесорний час, обсяг оперативної пам’яті, обсяг дискового простору, продуктивність підсистеми зберігання, продуктивність підсистеми мережевої взаємодії та ін. Споживання вказаних ресурсів змінюється залежно від багатьох чинників, які складно врахувати при виборі керуючих впливів [329]. Зазвичай, ВМ споживають не весь обсяг ресурсів, який замовлений при їх створенні і розгортанні. Максимальне споживання ресурсів виникає епізодично і не завжди одночасно для всіх ВМ, що розміщені на ФС. Тому, виникає можливість розміщувати більшу кількість ВМ на одному ФС, ніж передбачено максимальним споживанням ресурсів. Але при цьому, при виникненні пікових навантажень на більшість ВМ, наявних фізичних ресурсів ФС не вистачає, що призводить до збільшення затримки при відгуку на запити та збільшення часу виконання застосувань. При цьому порушується SLA для всіх ВМ на такому ФС.

Таким чином, виникає задача прогнозування споживання обчислювальних ресурсів для фізичного або віртуального сервера з метою зменшення кількості порушень SLA та зменшення енергоспоживання через прийняття управлінських рішень щодо розміщення нових ВМ та міграції існуючих.

4.3. Моделі прогнозування для управління IT-інфраструктурою

Для ефективного визначення стратегій управління ресурсами IT-інфраструктури хмарного ЦОД, ємністю, потужністю та консолідацією ВМ пропонується визначити чотири рівня прогнозування: прогнозування навантаження на ресурси ФС, прогнозування запитів користувачів до сервісів, прогнозування створення нових ВМ та прогнозування кількості завершених ВМ (ВМ, які вимикаються через нормальне завершення або збої). Перший рівень прогнозування впливає на прийняття рішення про вибір стратегії управління і подальше розміщення нової ВМ та міграцію існуючих ВМ. Інші рівні прогнозування впливають на процес прийняття рішень для вирішення задач в рамках визначеної стратегії, управління балансуванням навантаження та

управління продуктивністю, які в кінцевому підсумку теж впливають на процес прийняття рішень для вирішення задач консолідації ВМ і управління ресурсами.

Навантаження на корпоративні та публічні хмарні послуги має сезонні характеристики. Але сезонність притаманна тільки певним видам навантажень і в публічній хмарі, де навантаження постійно змішуються і обслуговуються одним і там самим обладнанням, виявити сезонність дуже складно. Крім того, в кожного сервісу може бути своя сезонна характеристика. Три інтервали прогнозування навантаження ЦОД запропоновані в літературі: короткострокові прогнози (на період приблизно від хвилини до десятків хвилин), середньострокові прогнози (на період приблизно декількох десятків годин) і довгострокового прогнозування (на період приблизно декількох днів). Інтервал короткострокового прогнозування зазвичай порівнянний з тривалістю процесів керування, такими як включення/вимикання ФС, створення і запуск ВМ, міграція ВМ та ін. Але час роботи ВМ може, також, вимірюватись сотнями годин. Наприклад, щоб включити ФС, зазвичай потрібно три-чотири хвилини [109].

У дисертаційній роботі проаналізовано декілька підходів до використання техніки прогнозування спільно з моделлю у просторі станів ЦОД. По-перше, розглянуто підхід до управління з прогнозуванням на основі моделі (MPC) для вирішення задачі управління балансуванням навантаження та управління продуктивністю. Переваги використання MPC добре відомі в літературі [331]. Підхід MPC дозволяє розрахувати впливи керування в межах обмежень системи, використовуючи прогнозовані значення споживання ресурсів ЦОД та дані зворотного зв'язку для обчислення неточності моделі та невизначеності системи. Але пошук оптимальних керуючих впливів $x_{ij}(t)$, $y_{ij}(t)$, $u_{ij}(t)$, $s_{ij}(t)$ і $S_{ij}(t)$ спільно з використанням запропонованої моделі динаміки ЦОД вимагає значних онлайн-обчислень для виконання оптимізації за рахунок великої кількості вхідних/вихідних змінних, обмежень та довжини горизонту прогнозу. Наприклад, з урахуванням 10 типів ВМ [105] і понад 20 тис. ФС, отримана модель ЦОД буде містити принаймні 200 тис. змінних, що ускладнює розв'язання задачі оптимізації в режимі реального часу. Однак можливість застосування MPC для управління ресурсами ЦОД потребує подальших досліджень.

По-друге, розглянуто можливість використання експоненці́йного згла́джування (EWMA) для прогнозування. EWMA - це метод математичного перетворення, який застосовує коефіцієнти зважування до членів часового ряду, які зменшуються експоненційно. Робоче навантаження та запити користувачів хмарних послуг можуть значно змінюватись і EWMA не враховує тренди зростання навантаження. Незважаючи на деякі недоліки використання EWMA для прогнозування навантаження на ресурси хмари, в деяких випадках може бути прийнята модифікація на основі EWMA, наприклад, запропонована в [5]. По-третє, розглянуто можливість використання регресійного аналізу [14], який широко використовується для короткострокового прогнозування навантаження. Моделі лінійної авторегресії можуть бути використані для прогнозування навантаження на ресурси хмарного ЦОД [2, 3]. Наприклад, за допомогою лінійної авторегресії кількість нових розгортань ВМ, яка залежить від значення іншої незалежної змінної, можна передбачити за допомогою рівняння лінійної регресії і історичних спостережень. У цьому випадку параметри моделі визначаються навчанням на історичних даних. Також, в публікаціях [4, 13] представлені результати досліджень короткострокового прогнозування навантаження з використанням сезонних часових рядів. У розділі 7 дисертаційної роботи проаналізовано роботу методу короткострокового прогнозування навантаження, що запропонований у [2]. Причиною цього є можливість включити сезонну структуру зміни робочого навантаження.

Прогнозоване число ВМ, яке необхідно включити і розмістити на ФС на наступному кроці управління ЦОД, позначено як $\hat{N}_{VM}^{on}(t)$ і визначено у такий спосіб:

$$\hat{N}_{VM}^{on}(t) = \sum_{k=1}^{n} a_k N_{VM}^{on}(t - kT) + \sum_{j=1}^{m} b_j \Delta N_{VM}^{on}(t - j),$$

$$\Delta N_{VM}^{on}(t - j) = N_{VM}^{on}(t - j) - \tfrac{1}{n} \sum_{k=1}^{n} N_{VM}^{on}(t - j - kT),$$

де a_k – це параметр, який враховує періодичну складову моделі протягом часового горизонту $\Upsilon_1 = \{1, 2, ..., T\}$ в термінах всіх попередніх вимірювань, b_j – це параметр локальної корекції для обліку кореляцій між найближчими

значеннями перед прогнозованим, n – глибина попередніх вимірювань для налаштування на періодичну складову, m – глибина попередніх вимірювань для врахування впливу найближчих вимірів на прогноз. Недоліком цього методу є те, що прогнозування сплесків практично неможливо, оскільки вони викликані відмовами ФС і змінними навантаженнями, а також їх сумішами.

У результаті аналізу часових інтервалів виконання керуючих впливів в умовах хмарних ЦОД і результатів експериментальних досліджень розділу 9 можна зробити висновок, що наведений метод прогнозування забезпечує достатньо точні прогнозні значення для запропонованого методу ІУР ЦОД при сталому режимі роботи ЦОД, коли коефіцієнт життєздатності ВМ (3.23) приблизно дорівнює 1. Але при інших стратегіях управління, що застосовуються в режимах відмінних від сталого, потрібно використовувати розроблений у розділі 4 комбінований метод прогнозування. Часовий інтервал, через який виконується наступний крок управління, визначається з урахуванням таких процесів: час міграції ВМ у ЦОД становить приблизно 10-40 с, час міграції ВМ між ЦОД становить приблизно 10-15 хвилин, час увімкнення ФС становить приблизно 3-4 хвилини, а дані моніторингу надходять до системи управління з інтервалом 1 хвилина або з інтервалом 5 хвилин для некритичних застосунків.

У даному пункті запропонована методика прогнозування $\hat{N}_{VM}^{on}(t)$ для сталого режиму роботи ЦОД. Інші змінні моделі ІУР, такі як кількість запитів користувачів на послуги та кількість вимкнених або створення нових ВМ, можна передбачити аналогічно.

4.4. Адаптивний метод прогнозування навантаження на ресурси IT-інфраструктури

4.4.1. Засади адаптивного методу прогнозування

Визначити розподіл вибірки та оцінити параметри і коефіцієнти моделі для прогнозування споживання ресурсів при будь-яких навантаженнях і їх комбінаціях, а також при різних стратегіях управління не уявляється можливим. Таким чином, потрібно через певний проміжок часу створювати нову модель, застосовуючи нові дані, що надійшли від підсистеми моніторингу. Розмір

навчальної вибірки, що потрібна для оцінки моделі прогнозу на поточному кроці управління, треба теж адаптувати до поточних умов функціонування хмарного ЦОД.

Оскільки строгого критерію застосування того чи іншого методу при прогнозуванні споживання обчислювальних ресурсів не знайдено, в дисертаційній роботі пропонується адаптивний метод з адаптацією параметрів моделі і комбінуванням прогнозів. Комбінований метод прогнозування навантаження на обчислювальні ресурси полягає у застосуванні декількох (альтернативних) методів прогнозування з підбором розміру навчальної вибірки і подальшим вибором прогнозу на основі визначеного критерію. Пропонуються такі критерії: мінімальна похибка прогнозу на попередньому(іх) кроках управління, усереднення прогнозів та зважене комбінування прогнозів. Склад альтернативних методів прогнозування визначається емпіричним шляхом за результатами попередніх досліджень їх застосування до навчальних вибірок типового навантаження. Основним фактором, що впливає на множину альтернативних методів, є кількість методів, результати їх попереднього застосування для типових навантажень (читання/запис даних з БД, веб-сервер, MapReduce, робота з ERP системою та ін.), а також час обчислення прогнозів кожним з них.

При розробленні методів для прогнозування динаміки досліджуваних процесів використано системний підхід, який ґрунтується на таких принципах системного аналізу: принцип системної координації функцій; принцип функціональної повноти та ортогональності функцій; принципи мультифакторної адаптації та раціонального доповнення. Задачі, які випливають з системного підходу до моделювання і прогнозування, ілюструє рис. 4.1 [26].

Твердження 1. Використання принципів системного аналізу у межах спеціалізованої системи прогнозування, яка містить функції ідентифікації і оброблення можливих невизначеностей (що зустрічаються у процесах моделювання і прогнозування), а також контроль обчислень на всіх етапах оброблення даних дають можливість обчислювати оцінки прогнозів високої якості.

Рис. 4.1. Задачі системного аналізу у прогнозуванні

4.4.2. Адаптивний метод прогнозування навантаження

Як вже було зазначено, проблема прогнозування потреби в ресурсах для ФС не є тривіальною. Об’єм фізичних ресурсів, які споживає ФС з працюючими ВМ, дорівнює сумі потреб у віртуальних ресурсах з боку цих ВМ. Таким чином, є два способи прогнозувати потреби фізичних ресурсів. Перший – використовуючи дані моніторингу споживання ресурсів ФС. Другий – використовуючи дані моніторингу споживання ресурсів всіма ВМ, що виконуються на ФС. Аналіз особливостей першого і другого способів наведений у табл. 4.1.

Табл. 4.1. Порівняння способів отримання прогнозу споживання обчислювальних ресурсів

Спосіб	Переваги	Недоліки	Обмеження
Прогнозування споживання ресурсів для ФС)	Одноразова побудова моделей прогнозу та одноразове обчислення прогнозу	Вхідні та вихідні міграції спотворюють реальне споживання ресурсів ВМ	Потрібно враховувати дані моніторингу, які отримані після завершення вхідних та вихідних міграцій
Прогнозування споживання ресурсів для всіх ВМ на ФС	Кількість історичних даних моніторингу кожної ВМ не залежить від кількості міграцій. Дані про споживання ресурсу (і попередні	Побудова моделей прогнозу та обчислення прогнозів виконується для кожної ВМ. Відбувається	Немає

	моделі з прогнозами) можуть мігрувати разом з ВМ.	накопичення помилки прогнозу в цілому для ФС.	

Потреба ресурсів для ВМ може значно змінюватися в часі під впливом числа клієнтських запитів і архітектури конкретних застосувань, які працюють всередині ВМ [92]. Розглянути всі можливі комбінації розміщення ВМ з конкретними потребами ресурсів не представляється можливим. Крім того, моделі прогнозу і їх параметри, як правило, повинні бути обрані відповідно до конкретних умов використання методу з дотриманням обмежень. У дисертації запропонований адаптивний метод прогнозування зі структурною адаптацією моделі прогнозу, який використовується для прогнозування навантаження на процесор ВМ. Це обумовлено використанням відповідних даних моніторингу роботи ВМ для експериментального дослідження запропонованого методу і моделей. Запропонований метод може бути використаний так само і для прогнозу потреби інших ресурсів ФС.

Оскільки умови роботи сервера (віртуального або фізичного) залежать від багатьох випадкових факторів, як зовнішніх, так і внутрішніх, необхідно адаптуватися до різних умов при прогнозуванні навантаження через використання структурної адаптації. У цьому випадку структурна адаптація полягає у тому, щоб вибрати кращий метод та кращу модель прогнозу (її параметри) залежно від поточних умов експлуатації сервера.

Прогноз споживання кожного ресурсу виконується окремо і не залежить від методів, обраних для прогнозування споживання інших ресурсів. Розглянемо роботу методу на прикладі прогнозування споживання процесорного ресурсу (структурна схема методу оцінювання моделі подана на рис. 4.2). Для оцінювання структури моделі розраховуються автокореляційна функція (АКФ), часткова АКФ, кореляційна матриця і коваріаційні функції. Оцінювання параметрів моделі може бути здійснено за такими методами: найменших квадратів (МНК), нелінійним МНК, методом максимальної правдоподібності і Монте-Карло для марковських ланцюгів.

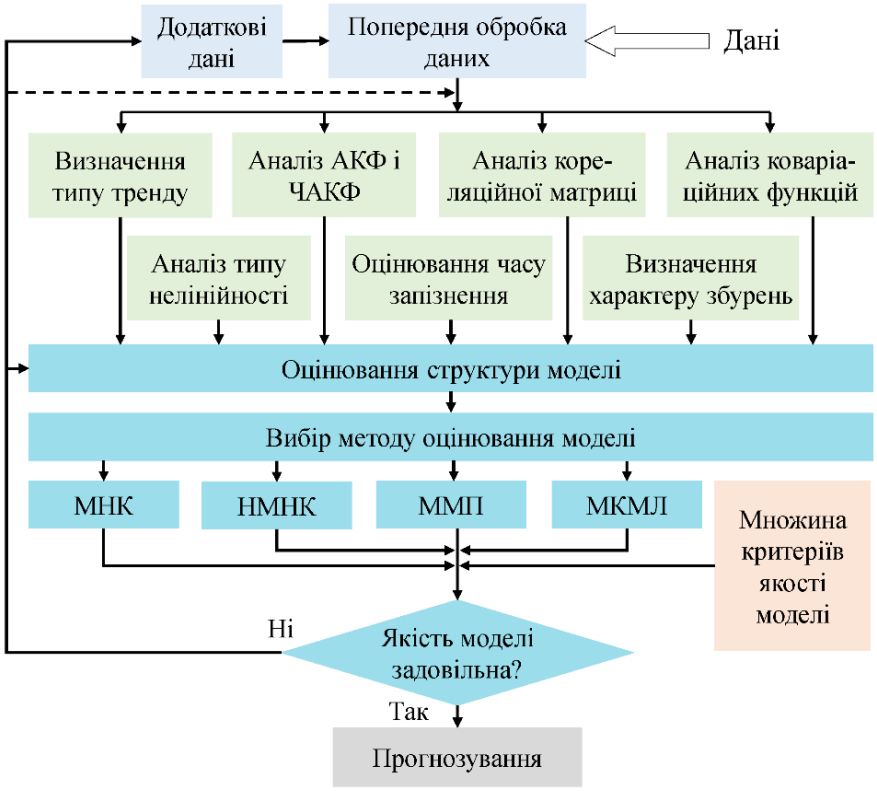


Рис. 4.2. Схема адаптивного методу оцінювання

Запропонований в дисертації метод використовує ідеї, представлені в роботі автора [16]. Схема адаптивного оцінювання прогнозу показана на рис. 4.3.

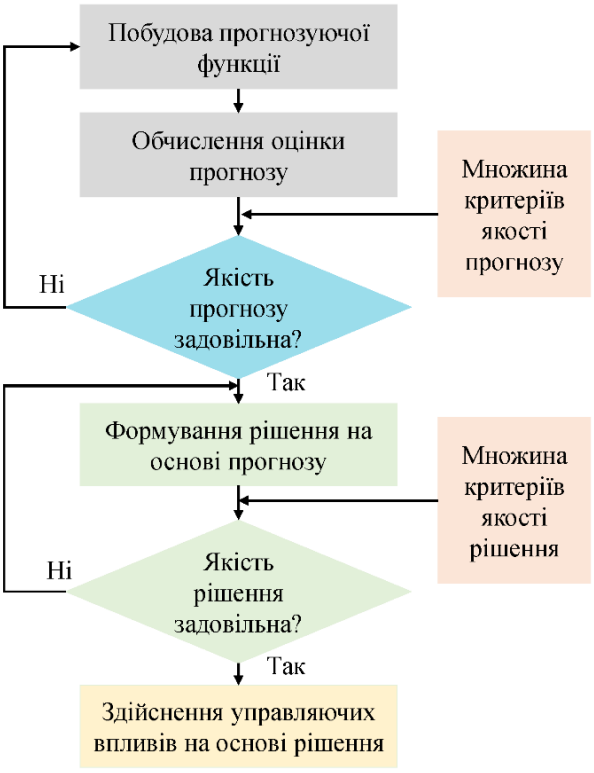


Рис. 4.3. Схема процедури адаптивного оцінювання прогнозу і формування рішення на його основі

На першому кроці роботи методу, серед обраних емпіричним шляхом альтернативних методів і моделей прогнозування, для прогнозування навантаження на поточному кроці використовується той метод/модель, що дав найкращий прогноз на попередньому кроці. Попереднім кроком назвемо стан системи, коли надійшли фактичні дані про споживання ресурсу і є можливість обчислити середню абсолютну похибку прогнозу в процентах (САПП). Таким чином, для вибору управління на поточному кроці береться прогноз, який отриманий методом, що дав мінімальну середню абсолютну похибку прогнозу в процентах на попередньому кроці. На другому кроці визначаються параметри моделей та їх структура. У дисертації використані вбудовані в R [17] алгоритми підбору параметрів моделей та розроблено новий алгоритм підбору розміру навчальної вибірки (або довжини вікна тренувальних даних). При цьому, розмір вікна тренувальних даних вибирається таким, з допомогою якого на попередньому кроці було отримано прогноз з мінімальною САПП. Робота першого і другого кроків методу базується на припущенні, що більш точний прогноз на наступному кроці буде отриманий тим методом і тією моделлю, за допомогою яких отриманий найкращий прогноз на поточному кроці. Обидва кроки роботи методу є адаптивними, на першому кроці до поточних умов адаптується метод, на другому кроці будується функція прогнозування і вибирається розмір навчальної вибірки даних цього методу.

Попереднє оброблення даних необхідне для приведення їх до форми, яка забезпечить можливості коректного застосування методів оцінювання параметрів моделі та отримання їх статистично значущих оцінок. Так, досить часто необхідно заповнювати пропуски даних, корегувати значні імпульсні (екстремальні) значення, нормувати значення у заданих межах, логарифмувати великі значення та фільтрувати шумові складові [286].

На основі коректно підготовлених даних оцінюються структури і параметри математичних моделей-кандидатів процесів, вибраних для прогнозування та керування. Вибір (оцінювання) структури моделі – ключовий момент її побудови. Нагадаємо, що структура моделі включає п’ять елементів: (1)

вимірність (число рівнянь, які утворюють модель); (2) порядок – максимальний порядок диференціальних або різницевих рівнянь, які входять в модель; (3) нелінійність та її тип (нелінійності відносно змінних або параметрів); (4) час затримки (лаг) реакції відносно входу та його оцінка; (5) зовнішнє збурення процесу та його тип (детерміноване або випадкове). Як правило, для одного процесу оцінюють декілька моделей-кандидатів, а потім вибирають з них кращу за допомогою множини статистичних параметрів якості моделі [286].

Отримані моделі використовуються для обчислення прогнозованого значення на *один крок* вперед. Зазвичай, для перевірки адекватності отриманої моделі необхідно проаналізувати залишки моделі (residuals). Якщо за допомогою автокореляційної функції і щільності розподілу встановлено, що залишки некорельовані і мають нормальний розподіл (або наближений до нього), то модель є адекватною.

Якість моделі оцінюють за допомогою декількох статистичних критеріїв якості, зокрема таких: коефіцієнта множинної детермінації (R^2), який характеризує інформативність моделі по відношенню до інформативності даних; статистики Дарбіна-Уотсона (*DW*) [26], що визначає ступінь автокорельованості похибок моделі; інформаційного критерію Акайке (*AIC*) і статистики Байєса-Шварца (*BSC*); суми квадратів похибок моделі ε, ($SSE=\sum\varepsilon^2(k)$); F – статистики Фішера та інших. Для автоматизованого вибору кращої моделі можна скористатись інтегральним критерієм якості [26] *IK*, який визначається як:

$$\mathrm{I}K = e^{1-R^2} + \frac{SSE}{N} + \begin{cases} \ln(AIC+BSC), & якщо\ \ AIC+BSC>0 \\ e^{AIC+BSC}, & якщо\ \ AIC+BSC\le 0 \end{cases} + e^{2-DW} + \\ + \ln(СКП) + \ln(САПП) + e^{U},$$

де $СКП$ – середньоквадратична похибка однокрокового прогнозу на навчальній (історичній) вибірці; $САПП$ – середня абсолютна похибка прогнозу в процентах; U – коефіцієнт Тейла (наближається до нуля, якщо модель придатна для прогнозування) [26].

В умовах автоматичного прогнозування навантаження визначити розподіл або статистичні характеристики навчальної вибірки не уявляється можливим.

Тренувальні дані (які у вітчизняній літературі також називають навчальною вибіркою) постійно змінюються, надходять від підсистеми моніторингу і використовуються для створення нової прогнозної моделі на кожному кроці управління. У запропонованому методі зроблено припущення, що неможливо визначити розподіл вибірки, обрати структуру і параметри моделі прогнозу навантаження, використовуючи тільки один фіксований розмір навчальної вибірки (набір послідовних спостережень) навантаження, отриманий в одній або декількох ВМ. Така неможливість обумовлена, зокрема, складністю отримання таких початкових даних, як: розмір навчальної вибірки, кількість навчальних вибірок, кількість і тип ВМ, з яких отримуються вибірки.

Таким чином, в запропонованому адаптивному методі прогнозування неможливо на кожному кроці аналізувати адекватність моделей прогнозу загальноприйнятим шляхом. Тому пропонується визначати точність отриманої моделі прогнозу за допомогою середньої абсолютної похибки в процентах (MAPE, [18]) не перевіряючи адекватність моделі і не обираючи найкращу модель серед обчислених. Використання метрики MAPE обумовлене необхідністю порівняння точності прогнозу, отриманого різними методами на різних навчальних вибірках даних. Якщо MAPE <10%, то прогноз отриманий з високою точністю, при 10%< MAPE <20% точність прогнозу добра, при 20%< MAPE <50% точність прогнозу задовільна, при MAPE >50% точність прогнозу незадовільна.

Базуючись на прогнозованих значеннях, менеджер ФС може приймати рішення про виділення більшого об’єму ресурсу для ВМ, або міграції ВМ в межах однієї стійки. Прогнозовані значення враховуються також на верхньому рівні управління для визначення стратегій управління ресурсами хмарного ЦОД.

4.4.3. Математична модель адаптивного методу прогнозування

У загальному вигляді задача прогнозування навантаження з використанням запропонованого адаптивного методу прогнозування сформульована так.

Представимо альтернативні конфігурації адаптивного методу прогнозування у вигляді системи $\langle A,W \rangle$, де A – множина альтернативних методів прогнозування, $|A| = n$, n – кількість елементів множини A, W – множина розмірів навчальних вибірок, $|W| = m$, m – кількість елементів множини W. Навчальна вибірка – це набір значень визначеного показника (прогнозованого параметру), який отриманий з системи моніторингу починаючи з попереднього кроку управління. Позначимо змінною $Q_{i,j} = f(a_i, w_j)$, $i = \overline{1,n}$, $j = \overline{1,m}$ середню абсолютну похибку в процентах, що отримана при використанні альтернативного методу a_i, $a_i \in A$, з розміром навчальної вибірки w_j, $w_j \in W$, $w_{\min} \le w_j \le w_{\max}$. Позначимо змінною $K_{t|t-1} = \{a_i, w_j, Q_{i,j}\}$ список конфігурацій адаптивного методу прогнозування, а змінною $K^p_{t|t-1} = \{a^p_i, w^p_j, Q^p_{i,j}\}$ – альтернативу p для вибору конфігурації адаптивного методу прогнозування, що буде обчислений на поточному кроці за допомогою конфігурації, що отримана на попередньому кроці t-1. Тоді в систему управління для прийняття рішень на поточному кроці t передається прогноз, отриманий за допомогою двійки $\{a_i, w_j\}_{t+1|t}$, яка визначається у такий спосіб:

$$\{a_i, w_j\}_{t+1|t} = \{a^p_i, w^p_j\}_{t|t-1} \Big| p = \arg\min_Q \{K_{t|t-1}\} \tag{4.1}$$

Список конфігурацій $K_{t|t-1}$, відсортований у порядку збільшення $Q_{i,j}$, позначимо як $C_{t|t-1}$. Таким чином, найкраща конфігурація для отримання прогнозу на поточному кроці t визначається як

$$C^1_{t|t-1} = K^p_{t|t-1} \Big| p = \arg\min_Q \{K_{t|t-1}\}. \tag{4.2}$$

Стадії роботи адаптивного методу прогнозування на поточному і попередньому кроках показані на рис. 4.4.

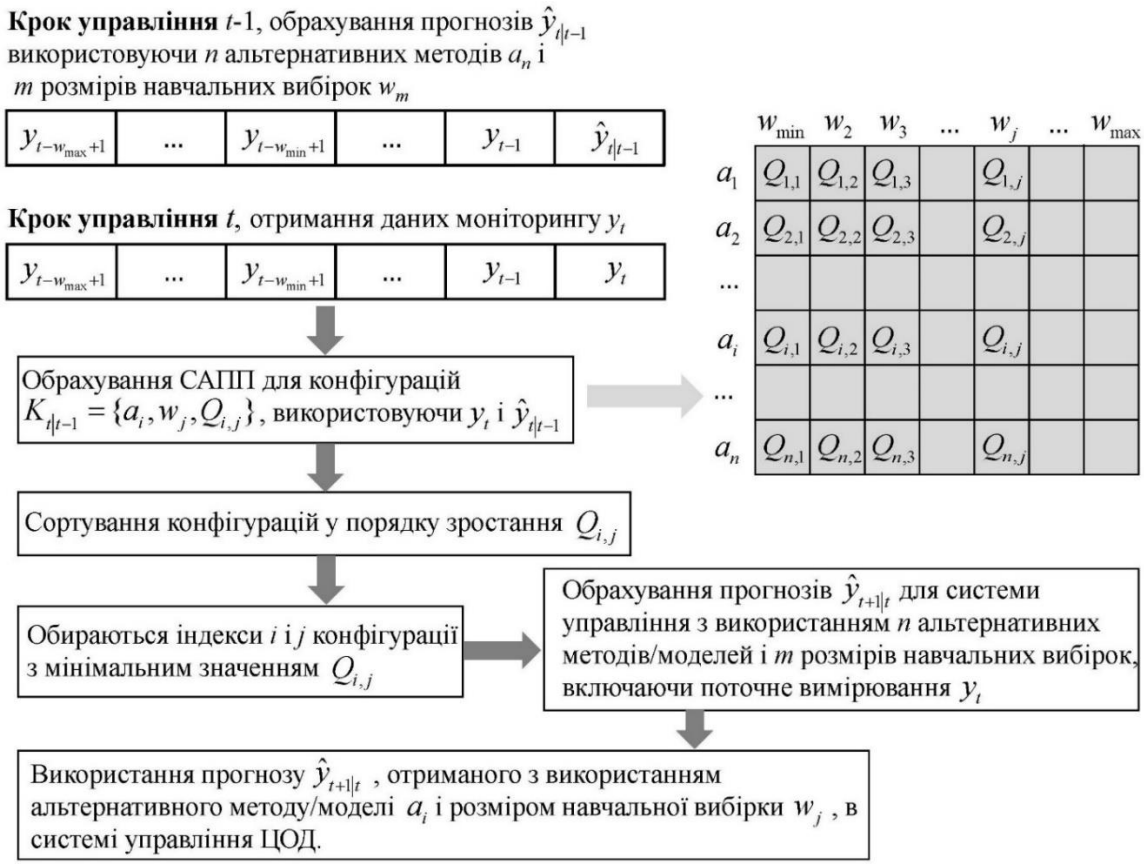


Рис. 4.4. Стадії роботи адаптивного методу прогнозування

4.5. Альтернативні методи і моделі прогнозування

Як зазначалося вище, в запропонованому адаптивному методі прогнозування склад альтернативних методів і моделей, що використовуються безпосередньо для обчислення прогнозу, визначений емпіричним шляхом. Кількість методів і їх тип пропонується підбирати залежно від типу ресурсу, споживання якого потрібно прогнозувати. У розділі 1 наведено опис альтернативних методів і моделей прогнозування для виконання порівняльного аналізу при застосуванні запропонованого адаптивного методу прогнозування. Ці альтернативні методи і моделі часто застосовуються в системах прогнозування, управління і моніторингу від провідних IT-компаній, а також в наукових публікаціях для прогнозування процесорного навантаження. При цьому доведено їх ефективність і можливість застосування при обмеженні на час виконання обчислень і пошуку параметрів моделі прогнозування [7, 8, 350].

Просте експоненційне згладжування

Оцінювання та вибір параметрів моделі SES і обчислення прогнозу на один крок згідно (1.1) виконується функцією ses(ts, h=1) в R, де ts – часовий ряд. При цьому ця функція вирішує нелінійну оптимізаційну задачу з підбору параметрів моделей, що будуються, і початкових значень моделі через мінімізацію суми квадратів похибок. Наступним кроком є отримання відносної похибки прогнозу MAPE у відсотках викликом функції accuracy(fitses$mean[1],testts) в R, де fitses$mean[1] – прогнозоване значення, testts – фактичне значення, отримане від підсистеми моніторингу. Метод простого експоненційного згладжування позначений в роботі як SES.

Метод Хольта та демпфірований метод тренду

Оцінювання та вибір параметрів моделі за методом Хольта згідно (1.2) і обчислення прогнозу на один крок виконується функцією holt(ts, h=1) в R, де ts – часовий ряд. Похибку прогнозу MAPE повертає функція accuracy(fitholt$mean[1],testts) в R, де fitholt$mean[1] – прогнозоване значення, testts – фактичне значення, отримане від підсистеми моніторингу.

Оцінювання та вибір параметрів моделі за методом демпфіруваного тренду Хольта згідно (1.3) і обчислення прогнозу на один крок виконується функцією holt(ts, damped=TRUE, phi = 0.9, h=1) в R, де ts – часовий ряд. Похибку прогнозу MAPE повертає функція accuracy(fitdholt$mean[1],testts) в R, де fitdholt$mean[1] – прогнозоване значення, testts – фактичне значення, отримане від підсистеми моніторингу. Тренд Хольта та демпфірований метод тренду позначені в роботі відповідно HOLT та DHOLT.

Модель авторегресії з інтегрованим ковзним середнім

В системах управління дані, отримані з підсистем моніторингу, представляються у вигляді часового ряду. Кількість елементів часового ряду, що розглядаються як навчальна вибірка (тренувальні дані) для створення моделі, визначається вимогами до адекватності моделі та точності прогнозу. Аналіз часових рядів, що отримані для ВМ з [22] показав, що умови стаціонарності для великих вибірок, як правило, не виконуються, а для часток цих вибірок іноді можуть виконуватися. Тому, застосування методів авторегресії (AR), ковзного

середнього (MA) або їх комбінації (ARMA) обмежені. Якщо для досліджуваного часового ряду виконуються умови стаціонарності, то для прогнозування можливо обрати одну з моделей (або їх комбінацію).

Оцінювання та вибір параметрів моделі ARIMA згідно (1.6), включаючи окремі випадки (AR, MA, ARMA) виконується автоматично одним з варіантів алгоритму Hyndman-Khandakar [23] через виклик функції auto.arima() в R. Функція auto.arima() для отримання моделі використовує тест на стаціонарність, мінімізацію AIC та метод максимальної правдоподібності. Побудова моделі авторегресії з інтегрованим ковзним середнім позначено в роботі як AutoARIMA.

Модель лінійної регресії з трендом

Оцінювання параметрів моделі лінійної регресії з трендом виконується автоматично через виклик функції tslm(ts ~ trend) в R [27], де ts – часовий ряд, trend – параметр функції. Похибку прогнозу MAPE повертає функція accuracy(fittslm$mean[1],testts) в R, де fittslm$mean[1] – прогнозоване значення, testts – фактичне значення, отримане від підсистеми моніторингу. Функція tslm() для оцінювання коефіцієнтів моделі використовує метод найменших квадратів. Модель лінійної регресії з трендом позначена в роботі як LR.

Метод TBATS

Оцінювання та вибір параметрів моделі TBATS виконується автоматично через виклик функції tbats(ts, biasadj=TRUE) в R, де ts – часовий ряд, biasadj – параметр функції. За допомогою функції forecast(tbatsmodel, h=1) в R, де tbatsmodel – модель TBATS, h – горизонт прогнозу, обчислюється прогнозне значення на один крок вперед. Похибку прогнозу MAPE повертає функція accuracy(fcTBATS[["mean"]],testts) в R, де fcTBATS[["mean"]] – прогнозоване значення, testts – фактичне значення, отримане від підсистеми моніторингу.

4.6. Методи комбінування прогнозів, обчислених альтернативними методами і моделями прогнозування

Ідея комбінування прогнозів, отриманих за альтернативними методами прогнозування запропонована в роботі [30]. У результаті досліджень [102]

показано, що середнє арифметичне декількох прогнозів є більш точним прогнозом, чим його складові, взяті окремо.

Метод усередненого комбінованого прогнозування

Позначимо змінною $\hat{y}^{p}_{t+1|t}$ прогноз, отриманий за допомогою конфігурації $K^{p}_{t|t-1}$, що обчислена згідно (4.1). Відповідно, прогнози, отримані за допомогою конфігурацій $C^{1}_{t|t-1}, C^{2}_{t|t-1}, C^{3}_{t|t-1}, \ldots, C^{k}_{t|t-1}$, позначимо як $\hat{y}^{1}_{t+1|t}, \hat{y}^{2}_{t+1|t}, \hat{y}^{3}_{t+1|t}, \ldots, \hat{y}^{k}_{t+1|t}$.

Комбінований прогноз, що обчислений з урахуванням *k* найкращих конфігурацій $C^{i}_{t|t-1}, i = \overline{1,k}$, обчислюється у такий спосіб:

$$\hat{y}^{c}_{t+1|t} = \frac{1}{k}\sum_{i=1}^{k} \hat{y}^{i}_{t+1|t}\,. \tag{4.3}$$

Якщо окремі прогнози незміщені (це повинен забезпечувати метод прогнозування), то комбінований прогноз також буде незміщеним. Похибка комбінованого прогнозу:

$$e^{c}_{t|t} = y_{t|t} - \hat{y}^{c}_{t|t-1} = \frac{1}{k}\sum_{i=1}^{k} e^{i}_{t|t}\,,$$

де $y_{t|t}$ – фактичне значення прогнозованої змінної. Дисперсія похибки комбінованого прогнозу:

$$\mathrm{var}\left[\frac{1}{k}\sum_{i=1}^{k} e^{i}_{t|t}\right] = E\left[\frac{1}{k}\sum_{i=1}^{k} e^{i}_{t|t}\right]^{2}.$$

Для *k*=2 дисперсія похибки комбінованого прогнозу обчислюється як

$$\mathrm{var}\left[\frac{1}{2}\sum_{i=1}^{2} e^{i}_{t|t}\right] = \frac{\sigma_1^2 + \sigma_2^2 + 2\rho\sigma_1\sigma_2}{4},$$

де ρ – коефіцієнт кореляції між похибками прогнозу. Якщо похибки прогнозування за двома моделями незалежні, тобто, $\rho = 0$, то остання формула спрощується так:

$$\sigma_c^2 = \frac{\sigma_1^2 + \sigma_2^2}{4}\,.$$

Таким чином, якщо дисперсії близькі за значеннями і незалежні, то дисперсія комбінованої похибки буде значно меншою будь-якої з двох дисперсій.

Але навіть при існуванні досить високої кореляції між похибками прогнозування дисперсія похибки комбінованого прогнозу буде меншою ніж дисперсія кожної моделі окремо. Наприклад, нехай $\sigma_1^2 = \sigma_2^2 = 81$ і $\rho = 0.9$, тоді $\sigma_c^2 = 76.95$.

Навіть в цій ситуації спостерігається зменшення дисперсії похибки прогнозу після усереднення оцінок, отриманих за двома моделями прогнозу. Однак, ситуація змінюється у випадку, коли дисперсії індивідуальних похибок сильно відрізняються. Наприклад, нехай $\sigma_1^2 = 100$, $\sigma_2^2 = 25$ і $\rho = 0.9$, тоді $\sigma_c^2 = 53.75$.

Таким чином, якщо дисперсії похибок сильно відрізняються, то просте усереднення результатів не потрібно робити. Висновок такий: просте усереднення можна застосовувати у випадках, коли дисперсії індивідуальних похибок прогнозування приблизно рівні або не дуже сильно відрізняються за своїми значеннями.

Якщо інформація щодо характеристик індивідуальних прогнозів відсутня, то можна присвоїти різні вагові коефіцієнти окремим прогнозам на основі суб'єктивних або експертних суджень:

$$\hat{y}_{t+1|t}^{c} = \lambda_1 \hat{y}_{t+1|t}^{1} + \lambda_2 \hat{y}_{t+1|t}^{2},$$

де λ_1, λ_2 - вагові коефіцієнти. Очевидно, що більші значення вагових коефіцієнтів необхідно присвоювати тим індивідуальним прогнозам, які мають меншу дисперсію похибок. При цьому для коректності обчислень необхідно, щоб виконувалась умова: $\lambda_1 + \lambda_2 = 1$.

Як правило, похибки прогнозів для конкретних моделей і процесів відомі, або їх можна визначити. Це дає можливість об'єктивно підійти до розв'язку задачі вибору вагових коефіцієнтів. Оскільки моделі, які дають менші суми квадратів похибок прогнозів, генерують якісніші прогнози, то логічно прийняти

цю міру за основу для визначення вагових коефіцієнтів. Позначимо суму квадратів похибок прогнозування (для історичного прогнозу глибиною *N*) через

$$sse = \sum_{j=1}^{N} e_{t|t}^2 .$$

Тепер можна записати вирази для вагових коефіцієнтів окремих прогнозів:

$$\lambda_1 = \frac{1/sse_1}{1/sse_1 + 1/sse_2}, \lambda_2 = \frac{1/sse_2}{1/sse_1 + 1/sse_2},$$

де sse_1, sse_2 – суми квадратів похибок для кожного з методів, що використовуються в даному випадку. Наприклад, для $sse_1 = 100$, $sse_2 = 25$ коефіцієнти дорівнюють $\lambda_1 = 0.2, \lambda_2 = 0.8$. Таким чином, об'єктивно присвоєно більший ваговий коефіцієнт точнішому методу прогнозування.

Метод зваженого комбінованого прогнозування

Позначимо як S_i суму квадратів різниць між дійсним значенням, отриманим з системи моніторингу, і прогнозованим значенням, отриманим *i*-м методом прогнозування. S_i визначається у такий спосіб:

$$S_i = \sum_{r=1}^{l} \left(y_r - \hat{y}_r^i \right)^2 ,$$

де l – кількість попередніх кроків управління, y_r – дійсне значення змінної, $r = \overline{1,l}$, $\hat{y}_r^i$ – прогнозоване значення змінної, отримане *i*-м методом/моделлю прогнозування з найменшою помилкою САПП, із застосуванням розміру навчальної вибірки w_j, $i = \overline{1,n}$, $j = \overline{1,m}$, на *r*-му кроці управління.

Найкращим серед усіх отриманих прогнозів вважається такий, похибка MAPE якого найменша. Відповідно, найкращий метод прогнозу і найкращий розмір навчальної вибірки на даному кроці будуть ті, за допомогою яких отриманий найкращий прогноз.

Позначимо як q_i вагу прогнозу, отриманого *i*-м методом на кроці управління *t*. Вага q_i визначається у такий спосіб:

$$q_i = \frac{1/S_i}{\sum_{k=1}^{n}(1/S_k)}.$$

Зважений комбінований прогноз $\hat{y}^{cw}_{t+1|t}$ в цьому випадку визначається за допомогою зважених прогнозів, отриманих за допомогою *n* конфігурацій *K*, $\{a_i, w_j\}$. Зважений комбінований прогноз визначається у такий спосіб:

$$\hat{y}^{cw}_{t+1|t} = \sum_{i=1}^{n} q_i \hat{y}^{i}_{t+1|t}. \quad (4.4)$$

Прогноз $\hat{y}^{i}_{t+1|t}$ для системи управління отримується при середньому значенні розміру навчальної вибірки для кожного з *n* методів/моделей, тобто $w^{i}_{t+1|t} = \sum_{k=1}^{l} w^{i}_{t-k|t} \Big/ l$, де i - метод/модель прогнозу, $w^{i}_{t-k|t}$ - розмір навчальної вибірки на попередніх кроках, з якою був отриманий найкращий прогноз. Інші прогнози з використанням інших розмірів навчальної вибірки і альтернативних методів обчислюються на поточному кроці з метою коригування вагових коефіцієнтів для майбутніх зважених прогнозів.

Реалізація методу усередненого комбінованого прогнозування і методу зваженого комбінованого прогнозування наведена у додатку Б.

4.7. Оцінка якості параметрів моделі прогнозування

Для підвищення якості визначення параметрів моделі прогнозування використаємо підхід, запропонований в роботі [352]. При використанні кількох методів параметричної ідентифікації $\pi_1, \pi_2, ..., \pi_k$ для оцінювання параметрів моделі прогнозу пропонується обчислювати оптимальні групові уточнені оцінки $\bar{a}_j = \sum_{r=1}^{k} w_{jr} \tilde{a}_{jr}, j = \overline{1, p}$, де $\tilde{a}_{jr}$ - елемент вектора оцінок $\tilde{\mathbf{A}}_r, r = \overline{1, k}$. Оптимальність оцінки $\bar{a}_j$ досягається при мінімумі її можливої дисперсії $\mathrm{D}\{\bar{a}_j\} = \sum_{r=1}^{R} \sum_{g=1}^{k} w_{jr} w_{jg} \mathrm{cov}^{j}_{rg}$, де $\mathrm{cov}^{j}_{rg} = \mathrm{cov}(\mathrm{a}_{jr}, \mathrm{a}_{jg})$. Оцінка $\bar{a}_j$, яка відповідає умові $\mathrm{M}\{(a_j - \bar{a}_j)(\tilde{a}_{jr} - \bar{a}_j)\} = 0, r = \overline{1, k}$, має мінімальну дисперсію серед усіх

оцінок. Враховуючи вищесказане, можна використати таку систему рівнянь для отримання вагових коефіцієнтів $w_{jr}, r=\overline{1,n}$:

$$\begin{cases} \sum_{r=1}^{k} w_{jr}(\mathrm{cov}_{rg}^{j} - \mathrm{cov}_{rk}^{j}) = 0, g = \overline{1,k-1} \\ \sum_{r=1}^{k} w_{jr} - 1 = 0 \end{cases}.$$

Елементи коваріаційних матриць $[\mathrm{cov}_{rg}^{j}], j=\overline{1,p}$ одержують, обробивши відповідними методами параметричної ідентифікації π_r, π_g вибірки вхідних даних, після чого на отриманих множинах $\{\tilde{A}_{rq}^{\sigma}\},\{\tilde{A}_{gq}^{\sigma}\}$ обчислюються вибіркові оцінки $\mathrm{cov}_{rg}^{j} = \frac{1}{l-1}\sum_{q=1}^{l}(\tilde{a}_{jrq}^{\sigma} - \overline{a}_{jr})(\tilde{a}_{jgq}^{\sigma} - \overline{a}_{jg})$ [352].

Висновки до розділу 4

1. У розділі розроблено адаптивний метод комбінованого прогнозування навантаження на обчислювальні ресурси хмарного ЦОД з використанням усередненого і зваженого комбінування прогнозів, отриманих альтернативними методами і моделями прогнозування, і з адаптацією розміру навчальної вибірки, які забезпечують меншу помилку прогнозу у порівнянні з використанням одного методу або моделі прогнозу, що отримана на навчальній вибірці фіксованого розміру.

2. Моделі і методи усередненого комбінованого прогнозування і зваженого комбінованого прогнозування дають можливість адаптуватися до поточного стану системи (ФС і ВМ) і визначити керуючі впливи з урахуванням прогнозних значень. Якісне прогнозування навантаження на обчислювальні ресурси дозволяє застосувати проактивний підхід до управління ІТ-інфраструктурою зі зменшенням енергоспоживання та зменшенням кількості порушень угоди про рівень обслуговування клієнтів. При дослідженні запропонованого комбінованого адаптивного методу використані такі альтернативні методи прогнозування: метод простого експоненційного згладжування, метод Хольта, демпфірований метод тренду, моделі авторегресії інтегрованого ковзного середнього, модель лінійної регресії, метод TBATS.

3. Адаптивний метод комбінованого прогнозування навантаження також використовується на всіх рівнях управління, від рівня ЦОД до рівнів ФС і ВМ, в схемах реалізації стратегій і методах оптимізації, наведених у розділі 5. Також, запропонований метод використовується у схемах реалізації всіх стратегій управління ресурсами хмарного ЦОД, при визначенні трендів зміни навантаження і при визначенні стратегій управління ЦОД.

РОЗДІЛ 5. КОМПЛЕКС СХЕМ РЕАЛІЗАЦІЇ СТРАТЕГІЙ УПРАВЛІННЯ ІТ-ІНФРАСТРУКТУРОЮ ХМАРНОГО ЦЕНТРУ ОБРОБЛЕННЯ ДАНИХ З ВИКОРИСТАННЯМ ПРОГНОЗУВАННЯ

У розділі розроблено схеми реалізації стратегій управління з використанням запропонованих в дисертації методів консолідації ВМ і управління ресурсами на основі моделі динаміки та методу ІУР хмарного ЦОД. Розроблено метод рівномірної консолідації ВМ з використанням ідеї імітації відпалу. З використанням алгоритму променевого пошуку розроблено двостадійний метод управління ресурсами хмарного ЦОД. З використанням алгоритму навчання з підкріпленням розроблено модель і метод динамічної консолідації і розміщення ВМ. Для управління потужністю хмарного ЦОД розроблено метод, який відрізняється урахуванням динаміки станів ЦОД та їх прогнозів. Для управління розподіленим дворівневим сховищем з реплікацією в ІТ-інфраструктурі хмарного ЦОД розроблено оригінальні моделі і методи управління, що враховують комбінацію метрик стану сховища і мережі передачі між вузлами. Розглянемо зазначені методи більш детально, одночасно охоплюючи консолідацію ВМ як комплексну задачу.

5.1. Постановка задачі рівномірної консолідації віртуальних машин з використанням ідеї імітації відпалу

З метою розробки схеми реалізації стратегії S_1 в дисертації поставлена задача рівномірної консолідації ВМ. У хмарному ЦОД протягом певного часу працюють M ФС, з неоднорідною конфігурацією. Кожен ФС споживає декілька ресурсів, такі як процесорний час, оперативна пам'ять, розмір і продуктивність сховища, пропускна здатність мережевого інтерфейсу. Усі ВМ на ФС теж використовують ті ж самі ресурси, але надані ФС. Проблема розміщення ВМ виникає, коли потрібно визначити, яку саме ВМ слід розміщувати на визначеному ФС. Цю проблему потрібно вирішити в дисертації.

Проблема розміщення ВМ на ФС складається з двох частин. Перша частина проблеми – це початкове розміщення нових ВМ з різними потребами ресурсів до відповідних ФС з певною ємністю ресурсів. Друга частина проблеми розміщення ВМ – це перерозподіл деяких працюючих ВМ на інші ФС через зміни навантаження або через недовантаженість ФС. Обидві частини проблеми розміщення ВМ можуть розглядатися як задача упаковки в ємності, де N елементів з різними властивостями повинні бути розміщені в M ємностях з метою мінімізації кількості використовуваних ємностей. Проблема розміщення ВМ відрізняється від класичної проблеми пакування в ємності, оскільки властивості елементів можуть бути змінені (динамічна проблема), і існують інші обмеження при розміщенні ВМ на ФС [195]. Наприклад, для забезпечення живої міграції ВМ без її простою рекомендується обмежити максимальну кількість одночасних міграцій з/на ФС. Проблему розміщення ВМ можна розглядати також як багатовимірну задачу упаковки в ємності, оскільки кожен елемент має більше двох властивостей.

Важливо відзначити, що для забезпечення гарантії відсутності порушень SLA під час міграцій ВМ гіпервізори обмежують максимальну кількість одночасних міграцій з/на ФС, а також на мережевий ресурс і на сховище даних [196]. На деяких робочих навантаженнях, які інтенсивно використовують оперативну пам'ять або на старому апаратному забезпеченні велика кількість одночасних міграцій (більше рекомендованого чи практично визначеного числа) може швидко вичерпати системний ресурс і призвести до зупинки надання послуги і простою.

Вирішення проблеми динамічної консолідації ВМ складається з чотирьох частин [194]: (1) визначення перевантаженого ФС, з якого необхідно мігрувати одну або більше ВМ; (2) визначення ненавантаженого ФС і міграція всіх ВМ з цього ФС з метою його перемикання в режим сну; (3) визначення ВМ, які необхідно перенести з перевантаженого ФС; і (4) знаходження нових ФС для ВМ, вибраних для міграції з перевантажених та недовантажених ФС, а також нових ВМ, що створюються. У результаті вирішення частини (3) проблеми, ГМ

визначає список ВМ від перевантажених та недовантажених ФС, які будуть розміщені унаслідок міграції.

Як показано в [197], хмарні ЦОД постійно отримують непередбачувану кількість запитів розгортання ВМ з плином часу. У роботі [308] запропоновано алгоритм високого рівня для розміщення нових та існуючих ВМ з урахуванням нових запитів на створення ВМ під час пошуку і прийняття рішення. Відповідно до алгоритму високого рівня [308], ГМ запускає окремий примірник процесу керування для кожного запиту з ФС або для запиту створення нової ВМ. Пропонується спочатку обслуговувати запит з перевантаженого ФС, після чого опрацьовувати запит на створення нових ВМ, а потім – запит з недовантаженого ФС. Унаслідок роботи алгоритму високого рівня, ГМ отримує набір ВМ G^{VM} для розміщення в ЦОД. ГМ, також, визначає набір G^{PM} – це набір відповідних ФС на які можуть бути розміщені нові і мігруючі ВМ.

Таким чином, необхідно розробити алгоритм розподілу ВМ з набору G^{VM} використовуючи мінімальну кількість ФС і *рівномірно* навантажити ЦП, ОП та мережевий інтерфейс для досягнення максимальної продуктивності та повноцінного використання ресурсів ЦОД.

5.2. Метод рівномірної консолідації віртуальних машин з використанням ідеї імітації відпалу

Відповідно до схеми реалізації стратегії S_1, динамічне рівномірне розміщення ВМ на основі алгоритму імітації відпалу представлено в цьому розділі, та у працях [277, 320]. Проблема консолідації ВМ можна розглядати як проблему упаковки в ємності з перемінними розмірами елементів та динамічним попитом на обчислювальні ресурси. Для її вирішення застосовується модифікація алгоритму імітації відпалу.

Імітація відпалу є загальноприйнятою комбінаторною методикою оптимізації [198]. Вона використовує критерій Metropolis [199] і розподіл Больцмана для прийняття нового рішення. Використовуючи цю методику, можливо оптимізувати простір рішень задачі консолідації ВМ. У результаті

експериментальних досліджень, використовуючи температуру T як контрольний параметр, можливо отримати карту розміщення ВМ (розміщення кожної ВМ на кожному ФС), що дає кращі значення цільової функції. Для використання методики імітації відпалу необхідно визначити конфігурацію системи, функцію отримання нової конфігурації (карту розміщення ВМ), цільову функцію для задачі оптимізації та максимальну кількість дозволених послідовних ітерацій, які не дають поліпшення порівняно з поточною отриманою методом конфігурації.

Карта розподілу ВМ MAP_{best}, набір ВМ G^{VM}, а також набір ФС G^{PM} складають конфігурацію системи при використанні імітаційного відпалу. Метою алгоритму є отримання кращої карти розподілу ВМ в термінах цільової функції. На кожному кроці роботи алгоритму карта розподілу ВМ змінюється і переходить в новий сусідній стан за допомогою функції пошуку. Кожна нова карта розподілу ВМ оцінюється через порівняння відповідного індикатора прийнятності з індикатором прийнятності поточної карти розподілу ВМ для прийняття рішення про прийняття або відхилення нової карти розподілу ВМ як поточної карти розподілу. У даній роботі для вирішення задачі розміщення ВМ приймається максимальне число дозволених послідовних ітерацій, які не дають поліпшення порівняно з поточною конфігурацією, що дорівнює 100.

Алгоритм 5.1 представляє псевдокод оптимізації з використанням імітації відпалу (OIB). Унаслідок роботи алгоритму можуть бути отримані близькі до оптимальних карти розміщення ВМ. Основна мета розробленого алгоритму OIB полягає в обчисленні карти розподілу ВМ і передачі її ГМ для виконання процесу міграції. Наступний запуск алгоритму OIB виконується після завершення всіх міграцій ВМ, передбачених попереднім запуском. Рішення повинно бути картою розподілу ВМ з мінімальною кількістю використаних ФС та з рівномірним використанням обчислювальних ресурсів кожного ФС (процесора, оперативної пам'яті, сховища та мережевого інтерфейсу) для досягнення максимальної продуктивності та повноцінного використання ресурсів хмарного ЦОД.

Новий сусідній стан визначається за допомогою запропонованих двох типів функцій пошуку. По-перше, розроблена і оцінена проста функція пошуку (проста ОІВ). Проста стратегія полягає в тому, щоб поміняти місцями (між ФС) дві різні випадково вибрані ВМ, навіть якщо ВМ вже були переставлені на попередніх кроках алгоритму. Другою стратегією (ОІВ з обмеженнями) є обмеження кількості міграцій з/на ФС. У цьому випадку існують два обмеження: (1) конкретні ВМ можуть бути перепризначені тільки один раз під час оптимізації і (2) кількість ВМ, що мігрують з/на кожен ФС, обмежена.

В якості вхідних параметрів алгоритм отримує набір ВМ G^{VM}, які потрібно розмістити, набір ФС G^{PM}, температуру T і максимальну кількість дозволених послідовних ітерацій, які не дають поліпшення карти розподілу ВМ, *MaxCount*. Карта розподілу *MAP* зберігає не тільки розміщення ВМ на ФС, яку обчислив алгоритм 5.1, але також стани ВМ і ФС. Для ОІВ з обмеженнями, кожен ФС може бути обмежений для виконання міграцій ВМ, тому обмеження на максимальну кількість одночасних міграцій з/на ФС може бути досягнуто. У рядках 2 і 3 початковий стан кожного ФС (*MigrationCount*) встановлюється рівним нулю. Кожна ВМ може перебувати в міграційному стані (*VM.isInMigrationSatate* = true), чи ні (*VM.isInMigrationSatate* = false). У рядках 4 і 5 початковий стан кожної ВМ встановлюється в значення false.

Функція *getInitialSolution*() повертає початкове розміщення ВМ, обчислене за допомогою евристики, наприклад BFD або FFD. У експериментах використовується евристика BFD з сортуванням всіх ВМ в порядку зменшення їх поточного використання ЦП. Початкове розміщення ВМ обчислюється відповідно до однієї з визначених стратегій. Для ОІВ з обмеженнями максимальна кількість одночасних міграцій з/на ФС обмежена. У той же час, початкове розміщення ВМ не враховує використання оперативної пам'яті та використання мережевого інтерфейсу.

Алгоритм 5.1. Оптимізація розміщення ВМ за допомогою ІВ

Input: G^{VM}, G^{PM}, T, $MaxCount$

Output: Близька до оптимальної мапа розміщення ВМ на ФС MAP_{best}

1 $iter \leftarrow 0$
2 **for each** PM **from** G^{PM} **do**
3 $PM.MigrationCount \leftarrow 0$
4 **for each** VM **from** G^{VM} **do**
5 $VM.InMigrationSatate \leftarrow$ false
6 $MAP_{best} \leftarrow getInitialSolution\left(G^{VM}, G^{PM}\right)$
7 $MAP_{cur} \leftarrow MAP_{best}$
8 **while** $iter < MaxCount$ **do**
9 $MAP_{new} \leftarrow swap\left(MAP_{cur}\right)$
10 $IB_{new} \leftarrow getEstimation\left(MAP_{new}\right)$
11 $IB_{cur} \leftarrow getEstimation\left(MAP_{cur}\right)$
12 $dev \leftarrow IB_{cur} - IB_{new}$
13 **if** ($T > 10^{-5}$ and $e^{dev/T} >$ random()) OR ($T < 10^{-5}$ and $dev > 0$) **then**
15 $MAP_{cur} \leftarrow MAP_{new}$
16 $IB_{cur} \leftarrow IB_{new}$
17 **end if**
18 **if** $IB_{new} < IB_{best}$ **then**
19 $MAP_{best} \leftarrow MAP_{new}$
20 $IB_{best} \leftarrow IB_{new}$
21 $iter \leftarrow 0$
22 **end if**
23 $iter$ ++
24 $T \leftarrow 0.99T$
25 **end while**

Щоб знайти більш збалансований і продуктивний розв'язок викликається функція *swap*(). Псевдокод функції *swap*() для простого алгоритму ОІВ представляє алгоритм 5.2. Псевдокод для функції *swap*() для алгоритму ОІВ з обмеженнями міграції представляє алгоритм 5.3. Унаслідок роботи функції *swap*() обчислюється новий стан (нова карта розподілу ВМ). Нова карта розподілу ВМ отримується через випадковий вибір двох різних ВМ і зміну їх ФС, перша мігрує на ФС другої ВМ і навпаки. Ця перестановка ВМ призводить до створення нової конфігурації і відображає основні операції виконання пакування для рівномірного завантаження ФС. Основною перевагою алгоритму простої

OIB є його простота для отримання нової карти розподілу. Як результат, під час процесу оптимізації можна проаналізувати більше станів. Однак головним недоліком алгоритму простої OIB є те, що він не враховує виробничі вимоги і генерує занадто багато міграцій ВМ для кожного циклу оптимізації.

З часом алгоритм 5.1 відвідує більше станів, і температура T падає. У високотемпературних станах алгоритму дозволяється вибирати нову карту розподілу, навіть якщо вона гірша за існуючу. У цій роботі цільова функція мінімізується, і коли ми говоримо, що деяка карта розподілу ВМ призводить до кращої цільової функції, ніж деяка інша карта розподілу, ми маємо на увазі, що карта розподілу ВМ має меншу величину для цільової функції. Функція *getEstimation*() використовується для обчислення цільової функції.

Цільова функцію для оптимізації розподілу ВМ позначена як IB і визначена у такий спосіб (5.1):

$$IB = \sum_{i=1}^{K} IB_i, \qquad IB_i = \frac{1}{CPU_i RAM_i NET_i}, \tag{5.1}$$

де IB_i – це індикатор дисбалансу i-го ФС, CPU_i, RAM_i, NET_i – це завантаження процесора, пам'яті та мережевого інтерфейсу i-го ФС, K – це кількість ФС, що використовуються у початковому стані.

Високі значення IB_i вказують на незбалансоване використання ЦП, ОП і мережевого інтерфейсу, тоді як значення, близьке до 1, вказує на хороший рівень балансування. Значення, близькі до 1, не рекомендуються, оскільки попит на ресурси в найближчому майбутньому може збільшитися, тому бажано зарезервувати деякі ресурси для запобігання порушень SLA. Розглядаються три види ресурсів (ЦП, ОП і мережевий інтерфейс) в цільовій функції. Однак підхід може бути розширений до будь-якої кількості ресурсів, що підтримуються середовищем моделювання.

Алгоритм 5.2. *swap*() функція (звичайний ОІВ без обмежень)

Input: *MAP*

Output: Нова мапа розміщення ВМ на ФС *MAP*

```
1  for each VM_first from MAP do
2     ipm = getIndexOfPM(MAP, VM_first)
3     for each random PM from MAP do
4        jpm = getIndexOfPM(MAP, PM)
5        if ipm<>jpm then
6           for each VM_second from PM_jpm do
7              if PM_jpm.isSuitableForVM(VM_first) and PM_ipm.isSuitableForVM(VM_second)
   then
9                    MAP.swapped(VM_first, VM_second)
10                   return MAP
11             end if
12          end for
13       end if
14    end for
15 end for
```

Алгоритм 5.3. *swap*() функція (ОІВ з обмеженнями)

Input: *MAP*

Output: Нова мапа розміщення ВМ на ФС *MAP*

```
1  for each random VM_first from MAP do
2     ipm = getIndexOfPM(MAP, VM_first)
3     if not PM_ipm.MaxMigrationLimit and not VM_first.isInMigrationSatate then
4        for each random PM from MAP do
5           jpm = getIndexOfPM(MAP, PM)
6           if ipm<>jpm and not PM_jpm.MaxMigrationLimit then
7              for each VM_second from PM_jpm do
8                    if     PM_jpm.isSuitableForVM(VM_first) and
9                           PM_ipm.isSuitableForVM(VM_second) and
10                          not VM_second.isInMigrationSatate  then
11                                 VM_first.InMigrationSatate ← true
12                                 VM_second.InMigrationSatate ← true
13                                 PM_ipm.MigrationCountInc
14                                 PM_jpm.MigrationCountInc
15                                 MAP.swapped(VM_first, VM_second)
16                                 return MAP
17                   end if
18             end for
19          end if
20       end for
21    end if
22 end for
```

У [194] запропоновано дві метрики для вимірювання рівня порушень SLA в хмарі IaaS: відсоток часу, протягом якого активні ФС зазнали 100% використання ЦП, тобто час порушення SLA кожним ФС (*SLATAH*), що визначена рівнянням (5.2); загальне падіння продуктивності ВМ через міграції, тобто погіршення продуктивності внаслідок міграції (*PDM*), що визначено рівнянням (5.3). Автори [194] прийшли до висновку, що якщо ФС, що обслуговує застосунки, завантажений на 100%, продуктивність розміщених застосунків обмежена потужністю ФС, отже, продуктивність роботи ВМ знижується.

$$SLATAH = \frac{1}{M}\sum_{i=1}^{M}\frac{T_{S_i}}{T_{a_i}}, \tag{5.2}$$

$$PDM = \frac{1}{N}\sum_{j=1}^{N}\frac{C_{d_j}}{C_{r_j}}, \tag{5.3}$$

де M – це кількість ФС; T_{S_i} – це загальний час, протягом якого i-й ФС завантажений на 100%, що призводить до порушення SLA; T_{a_i} – це загальний час роботи i-го ФС; N – це кількість ВМ; C_{d_j} – це оцінка зниження продуктивності роботи j-ї ВМ, що викликана міграціями; C_{r_j} – загальна потужність процесора, яку вимагає j-та ВМ протягом своєї роботи; C_{d_j} оцінюється як 10% від використання ЦП в MIPS під час всіх міграцій j-ї ВМ [194].

Іншою метрикою, запропонованою в [194], є комбіноване порушення SLA (*SLAV*), яке охоплює як зниження продуктивності внаслідок перевантаження ФС, так і внаслідок міграцій ВМ (5.4). Об'єднана метрика, яка фіксує як споживання енергії, так і рівень порушень SLA, позначається як перевищення енерговитрат та порушення SLA (*ESV*) (5.5).

$$SLAV = SLATAH \cdot PDM, \tag{5.4}$$

$$ESV = E \cdot SLAV. \tag{5.5}$$

У дисертації розглянуто випадок, коли не виявлено перевантажених ФС. Таким чином, можливо використати політики управління з метою автоматичної генерації керуючих впливів. У деяких випадках протягом певного тривалого періоду часу є лише недовантажені ФС, і не було створено нових ВМ. Якщо

кількість невикористаних ресурсів ФС перевищує порогове значення, то ФС додається до списку консолідації G^{PM}. Алгоритм ОІВ перезапускається, якщо загальна кількість невикористаних ресурсів ФС, включених до списку консолідації, перевищує ресурси окремо взятого ФС. Усі міграції, ініційовані ОІВ на попередньому етапі, повинні бути завершені. Вибір порогового значення в даній роботі не розглядається і залежить від конфігурації хмарного ЦОД.

5.3. Модель двостадійного методу управління ресурсами хмарного центру оброблення даних на основі алгоритму променевого пошуку

На теперішній час більшість послуг із сфери інформаційного обслуговування клієнти отримують на базі хмарних ЦОД. Хмарні ЦОД являють собою складні системи, що складаються з серверних підсистем, підсистем зберігання даних, мережевих підсистем та підсистем інженерного забезпечення.

Розглянемо задачу управління ресурсами окремої підсистеми ЦОД у вигляді кластера з використанням віртуалізації. Для побудови кластеру в сучасних ЦОД використовують два підходи компоновки: з використанням гетерогенної конфігурації або гомогенної конфігурації. Обидва підходи мають свої недоліки та переваги, однак уникнути розміщення гетерогенних конфігурацій в масштабі ЦОД в більшості випадків не вдається. Конфігурації кожного кластеру можуть відрізнятися з причин еволюції елементної бази серверів, сховищ та мережевих пристроїв, а також появи нових вимог користувачів до IT-інфраструктури. Однак має місце і найгірший випадок, коли конфігурації ФС в кластері відрізняються [280].

Дослідження роботи запропонованого двостадійного методу [278, 280, 285], що базується на алгоритмі променевого пошуку, виконані для кластеру, який складається з ФС різної конфігурації. Кожен ФС надає для локальних ВМ такі ресурси як: процесорний час (CPU), об’єм пам’яті (RAM), доступ до підсистеми зберігання даних (IOPS), доступ до мережевої підсистеми (NET) та інші. Залежно від наявної ємності ресурсів ФС, вимог до ресурсів з боку ВМ, часу виконання завдань всередині ВМ та інтенсивності надходження завдань кількість ВМ, що виконуються на ФС, постійно змінюється. Зміна кількості локальних ВМ

відбувається в таких станах: розгортання нової ВМ, завершення роботи ВМ, міграція ВМ [280].

У сучасних умовах, при використанні промислових гіпервізорів та швидкісних локальних мереж, середній час міграції ВМ в середині ЦОД складає приблизно пів хвилини, залежно від обсягу пам’яті ВМ і інтенсивності її використання.

Крім того, на процеси управління ресурсами також впливає час включення ФС, який складає 3-4 хвилини залежно від конфігурації. Таким чином, краще перемикати ФС в режим сну. Переключення ФС з режиму сну в режим роботи відбувається значно швидше, чим холодний старт сервера.

Аналіз роботи запропонованого методу виконано з урахуванням двох ресурсів, що надаються для ВМ: CPU та RAM. Однак метод може бути доповнений іншими ресурсами, які потребує ВМ. Вибір саме двох ресурсів обумовлено використанням вхідних даних з набору Google cluster-usage traces (GCT) [106]. Для аналізу роботи алгоритму променевого пошуку використано дві таблиці з набору GCT, а саме "Machine events" та "Task events table". Випадковим чином з першої таблиці обрано 6000 ФС, з другої таблиці обрано 70000 завдань. З таблиці "Machine events" для кожного ФС використано такі атрибути: machine ID, capacity: CPU, capacity: memory. З таблиці "Task events table" для кожного ФС використано такі атрибути: "task index within the job", "machine ID", "resource request for CPU cores", "resource request for RAM" [280]. Дані в таблицях нормовані відносно ФС з найбільшим значенням ємності відповідного ресурсу серед ФС кластеру.

Процеси циклу управління можуть повторюватись через певні проміжки часу при наявності двох вимог: міграції ВМ, визначені на попередньому кроці завершені та є передумови для переключення ФС в режим сну. Таким чином, запропонований двостадійний метод управління ресурсами використовує потоки управління, що працюють *послідовно* [280].

Кластер управління складається з множини P з M ФС та множини V з N ВМ, $N, M \in \mathbb{N}^+$. У процесі розроблення та аналізу роботи алгоритму променевого

пошуку та у процесі циклу управління міграціями кількість ФС та ВМ не змінюється.

Для кожного завдання з таблиці "Task events table" розгортається окрема ВМ. У загальному випадку, між запусками алгоритму променевого пошуку кількість ФС та ВМ може змінюватись [280].

Задана ємність *j*-ї ВМ для ресурсу *k*, що позначена як $c_j^k \in (0,1]$, $k \in \{CPU, RAM\}$, визначається вимогами завдання і нормовано відносно ФС з найбільшою ємністю ресурсу *k*. Ємність ФС *i* для ресурсу *k*, що позначена $C_i^k \in (0,1]$, визначається типом ФС і нормовано відносно ФС з найбільшою ємністю ресурсу *k*.

У більшості випадків запропонована модель [280] передбачає зміни c_j^k та C_i^k протягом усього терміну дії ВМ і ФС. Проте модель не враховує зміни цих параметрів під час міграції ВМ або протягом кроку управління. У експериментальному дослідженні обмеження на параметри c_j^k та C_i^k накладаються вхідними даними, які отримані з наборів даних експлуатації кластера Google (GCT) [106]. Таким чином, змінні c_j^k та C_i^k зафіксовані під час моделювання.

Множина *P* складається з множини *A* ФС, які визначені для вимикання, та множини *B* ФС, які надають ресурси для ВМ, що будуть мігрувати з ФС, які належать до множини *A*, $A \bigcup B = P$, $A \bigcap B = \varnothing$.

Міграцію ВМ *j* на ФС *i* позначимо як $U_{ij} \in \{0,1\}$. Міграція відбувається якщо $U_{ij} = 1$. Кожна ВМ з множини *V* має свій ID, який у процесі роботи алгоритму пов'язаний з номером *j*. Кожний ФС теж має свій ID, який у процесі роботи алгоритму пов'язаний з номером *i*.

Основною метою розроблення і застосування двостадійного методу управління ресурсами хмарного ЦОД є зменшення кількості міграцій ВМ під час циклу управління та збільшення кількості ФС, що переключені в режим сну [280, 285].

5.4. Двостадійний метод управління ресурсами на основі алгоритму променевого пошуку

Двостадійний метод управління ресурсами хмарного ЦОД реалізує стратегію S_2 і забезпечує зменшення кількості міграцій ВМ під час циклу управління та збільшення кількості ФС, що переключаються в режим сну. В основу метода покладено алгоритм променевого пошуку та евристики для оцінки стану кластера ЦОД. Розроблення і дослідження методу представлені у працях [280, 285, 318, 323]. Робота методу складається з двох стадій. На першій стадії відбувається підготовка даних, на другій стадії визначається план міграцій ВМ.

Вхідними даними алгоритму променевого пошуку є: n – ширина променя, A – список ФС, які є претендентами для перемикання в режим сну, B – список ФС для обміну ВМ, в якому є вільні ресурси і є можливість розмістити додаткові завдання у вигляді ВМ [280].

Ідея алгоритму: перегляд i-го ФС зі списку A та пошук таких або такого ФС зі списку B, куди можливо мігрувати j-ту ВМ з i-го ФС. Якщо вдалося звільнити всі або частку ФС зі списку A, видаємо результат у вигляді матриці U_{ij}, яка є планом міграцій [280].

Опис роботи двостадійного методу променевого пошуку.

Перша стадія: підготовка вхідних даних для другої стадії. Формування списку A ФС, які треба звільнити від ВМ, та списку B ФС для визначення плану міграцій.

Друга стадія виконується для кожного ФС зі списку A:

1. На кожному кроці обираємо одну ВМ, назначену на i-й ФС і розглядаємо такі варіанти обміну (міграцій):

1) мігрувати ВМ на інший ФС, в якого залишилось достатньо вільних ресурсів CPU та RAM;
2) перевірити можливість обміну з іншим ФС віртуальною машиною, яка вимагає менше ресурсів (так ми розглядаємо лише стани з кращою оцінкою та уникаємо можливих зациклювань) і, у результаті, ФС не буде перенавантаженим після обміну [280].

2. З усіх можливих обмінів обирається *n* обмінів з найвищою оцінкою.

3. Завершуємо пошук якщо *i*-й ФС вдалося звільнити від ВМ або якщо не можна побудувати нові стани (тобто не залишилось жодного варіанту для реалізації допустимого обміну a) або b)).

Порівняння станів виконується за допомогою формули (5.6):

$$J = \sum_{i=1}^{m} u_i + \sum_{i=1}^{n} f_i \ , \tag{5.6}$$

де u_i – кількість використовуваних ресурсів на *i*-му ФС, f_i – кількість вільних ресурсів на *i*-му ФС, *m* – кількість ФС в списку *B*, *n* – кількість ФС в списку *A*.

Таким чином, алгоритм поступово зменшує використовувані ресурси на ФС, які належать до списку *A*, та завантажує ФС зі списку *B*.

Умови завершення алгоритму

Для завершення роботи алгоритму перевіряються такі умови:

1. Вичерпання списку *A*.

2. Неможливість мігрувати всі ВМ з певної визначеної кількості ФС Th_A поспіль [280].

Якщо другу умову не застосовувати, цикли пошуку виконуються занадто довго, порівняно з часом на створення нової ВМ та часом на міграцію для ВМ з середніми вимогами до ресурсів пам’яті. Для визначення Th_A пропонується застосувати евристику, яка полягає у розгляді певного відсотку ФС зі списку *A*, але не менше, ніж α ФС. При реалізації досліджуваного алгоритму прийнято $Th_A = 0.05$, $\alpha = 10$. З меншими значеннями α алгоритм пропускає значну кількість ФС, що могли бути переключені в режим сну, але унаслідок завершення алгоритму були не звільнені від локальних ВМ. Даний критерій завершення вводиться для зменшення часу виконання алгоритму.

Опис першої стадії роботи методу.

Отримання списку ФС, з яких треба мігрувати всі ВМ з метою подальшого переключення ФС в режим сну пропонується здійснювати за допомогою двох методик: *нижньої границі* та *порогу вільних ресурсів*. Розглянемо кожну з методик більш детально.

Методика нижньої границі полягає у визначенні такої кількості ФС, які вимкнути унаслідок міграції усіх ВМ, що на них працює, не уявляється можливим. Пошук нижньої границі виконується у такий спосіб:

1. Знаходимо середній об’єм наявних ресурсів по всім ФС, на яких працюють ВМ, $T_{av}^k = \frac{1}{Q}\sum_{i=1}^{Q} C_i^k, 0 < Q \le M$. Значення для кожного ресурсу k розраховуються окремо.

2. По кожному ресурсу окремо рахуємо суму необхідних ресурсів для виконання всіх наявних ВМ, $D^k = \sum_{j=1}^{R} c_j^k, 0 < R \le N$.

3. Окремо по кожному ресурсу рахуємо відношення необхідних ресурсів до середнього об’єму наявних ресурсів та округляємо значення до більшого цілого, $E^k = \lceil D^k / T_{av}^k \rceil$.

4. Нижньою границею буде найбільше число E^k з отриманих на кроці 3.

5. Сортуємо ФС одним з таких способів:

1) за об’ємом ресурсів ФС, потім за відношенням використаних ресурсів до кількості працюючих ВМ;
2) за об’ємом ресурсів ФС, потім за кількістю назначених завдань, потім відношенням використаних ресурсів до кількості локальних ВМ.

Унаслідок досліджень роботи алгоритму з’ясувалося, що варіант 2) виявився значно ефективнішим. При його використанні на одному й тому самому наборі даних вдалося вимкнути в середньому 82 ФС, тоді як при використанні варіанту а) вдалося вимкнути в середньому лише 33 ФС.

У результаті, знаходимо різницю між кількістю ФС, що використовуються, та отриманою нижньою границею. Знайдене число – це кількість ФС зі списку *A*, призначених для перемикання в режим сну. Ці сервери розташовані першими у відсортованому списку. Решта ФС потрапляє у список *B*. Таким чином, методика нижньої границі дозволяє отримати оцінку наявної вільної ємності фізичних ресурсів кластеру.

Ідея методики порогу вільних ресурсів полягає в такому формуванні списку A, при якому до цього списку потрапляють ФС, загальна кількість невикористаних ресурсів яких перевищує об'єм ресурсів одного з ФС кластеру.

Позначимо поріг вільних ресурсів як β. Побудова списку A з використанням порогу вільних ресурсів виконується у такий спосіб:

1. Встановлюємо значення β.

2. З множини P відбираємо ФС, у яких кожного вільного ресурсу більше за значення β.

3. По кожному ресурсу окремо рахуємо суму вільних ресурсів, $F^k = \sum_{i=1}^{Q} C_i^{k,free}, 0 < Q \le M$.

4. Сортуємо обрані ФС аналогічно до варіанту b) з методики нижньої границі.

5. Проходимо по відсортованому списку. Поки сума ресурсів більша за об'єм ресурсів поточного ФС віднімаємо від суми цей об'єм, додаємо поточний ФС в список A та переходимо до наступного ФС, інакше завершуємо проходження списку. Таким чином, отримуємо список A з ФС, які треба переключити в режим сну, та список B з ФС для обміну ВМ [280].

Реалізація двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку у вигляді діаграм класів застосунку моделювання наведена у додатку А.

5.5. Модель і метод динамічної консолідації і розміщення віртуальних машин на основі алгоритму навчання з підкріпленням

Розроблений в дисертації метод динамічної консолідації і розміщення ВМ хмарного ЦОД з використанням навчання з підкріпленням (НП, англ. reinforcement learning, RL) [206] реалізує стратегії S_5 та S_6 і забезпечує зменшення кількості порушень SLA, кількості увімкнутих ФС та кількості міграцій ВМ через навчання на результатах попередніх дій в певних станах системи з урахуванням прогнозованого навантаження. Основні результати розроблення і дослідження моделей і методів управління ресурсами IT-інфраструктури з

використанням НП представлені у працях [281, 324, 325, 330]. НП – один із способів машинного навчання, в ході якого програмні агенти виконують дії в певному середовищі з метою максимізації сукупної винагороди як результату дій. У дискретні моменти часу агент отримує дані спостереження, яке зазвичай включає і винагороду, та обирає керуючий вплив з множини доступних впливів на середовище. Середовище переходить до нового стану, і визначається винагорода, пов'язана з переходом в новий стан.

Метою агента НП є отримання якомога більшої винагороди. Агент може обирати будь-який вплив або згідно політики або випадково. НП є добре пристосованим для задач, які включають компроміс між довготерміновою та короткотерміновою винагородою. НП відрізняється від стандартного навчання з учителем тим, що пари правильних входів/виходів ніколи не представляються, а неоптимальні впливи (дії) явно не виправляються [281].

5.5.1. Задача консолідації віртуальних машин на основі алгоритму навчання з підкріпленням

В пункті 3.5 запропонована модель ЦОД як постачальника віртуалізованих ресурсів, який складається з m ФС з різною конфігурацією. Кожен ФС має процесор, який може бути багатоядерним, при цьому продуктивність вимірюється в мільйонах інструкцій на секунду. Крім того, ФС характеризується обчислювальною потужністю процесора, об'ємом оперативної пам'яті, пропускною здатністю мережі та показниками запису/читання жорсткого диску. Користувачі відсилають завдання або запити для створення n ВМ. Спочатку ВМ розміщуються на ФС відповідно до запитуваних характеристик. Відсоток використання ресурсів ВМ з часом буде змінюватись. Необхідно розмістити ВМ на мінімальній кількості ФС щоб мінімізувати витрати електроенергії та водночас не порушувати умови угоди про надання сервісів SLA.

Для управління віртуалізованими ресурсами ЦОД автори пропонують безмодельний варіант агента, що заснований на методі навчання з підкріпленням Q-learning [209]. Агент НП може отримати розв’язок близький до оптимального

через взаємодію з динамічним середовищем без будь-якої інформації про це середовище.

Спочатку ВМ розгортаються на ФС відповідно до запитуваних характеристик за допомогою алгоритму розміщення ВМ, що розроблений в [16].

Структура методу НП складається з:

– простору станів S – сукупність станів середовища, які агент може сприймати;

– простору дій A – сукупність дій, які агент може виконати.

– винагорода p_t, яка є або позитивною реакцією середовища, або штрафом (негативна реакція середовища) за виконану агентом дію. Таким чином, метою агента є зведення до мінімуму середніх довгострокових штрафів впродовж процесу навчання.

Q-learning – це один з найпопулярніших методів НП, який застосовується у багатьох галузях досліджень. На кожній ітерації алгоритму *Q*-learning агент отримує дані про поточний стан середовища $s_t \in S$ та вибирає дію $a_t \in A$, що впливає на середовище. Після виконання дії середовище переходить до наступного стану $s' \in S$, і агент отримує штраф p_t. На початку наступної ітерації t+1 агент спостерігає поточний стан системи $s_{t+1} \in S$, оновлює *Q*-значення кроку t на основі рівняння:

$$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha[p_t + \gamma \min_{a \in A} Q(s_{t+1}, a_{t+1})], \quad (5.7)$$

де $Q(s_t, a_t)$ – очікуваний довгостроковий штраф за виконання дії $a_t \in A$ в стані $s_t \in S$; α – швидкість (темп) навчання (англ. learning rate), коефіцієнт, що показує на скільки швидко нові дані про стани будуть враховуватись на наступних кроках, $\alpha \in [0,1]$; γ – коефіцієнт знецінювання (англ. discount factor), коефіцієнт, що визначає важливість майбутніх штрафів, $\gamma \in [0,1]$; $\min_{a \in A} Q(s_{t+1}, a_{t+1})$ – оцінка майбутнього *Q*-значення після виконання керуючого впливу $a_{t+1} \in A$ at t+1 step.

Якщо $\alpha = 0$, то агент не навчається, якщо $\alpha = 1$, то агент використовує дані про найновіші управляючі впливи. Якщо $\gamma = 0$, то агент розглядає лише поточні

винагороди, якщо $\gamma = 1$, то агент прагне до довгострокового мінімального штрафу.

Коли агент знову отримає стан $s_t \in S$, він вибере дію з мінімальним *Q*-значенням. Політика π вибору найкращої дії у стані $s_t \in S$ вираховується таким рівнянням:

$$\pi(s_t) = \arg\min_a (Q(s_t, a)). \quad (5.8)$$

Таким чином, метою агента НП є знаходження оптимальної політики відображення $S \to A$.

Алгоритм *Q*-learning має такі етапи:

Крок 1. Для кожної пари $s \in S$ та $a \in A$, ініціалізувати *Q*-значення, що дорівнює нескінченності;

Крок 2. Отримати поточний стан $s_t \in S$;

Крок 3. Обрати дію $a_t \in A$ випадковим чином або за допомогою (5.8);

Крок 4. Виконати дію $a_t \in A$;

Крок 5. Отримати винагороду p_t;

Крок 6. Отримати новий стан $s' \in S$ та оновити $Q(s_t, a_t)$ за допомогою (5.7);

Крок 7. $s \leftarrow s'$;

Крок 8. Повернутися до кроку 2.

Існує два способи вибору дії з простору можливих дій у кожному стані:

– вибір випадкової дії може відбуватися на початку навчання, коли оптимальні дії ще не відомі;
– дія вибирається відповідно до вивченої політики π.

5.5.2. Модель управління ресурсами IT-інфраструктури на основі алгоритму навчання з підкріпленням

Модель хмарного ЦОД для розроблення методу управління на основі НП показана на рис. 5.1. Структура складається з набору ФС, кожний з яких характеризується апаратною та операційною платформами і має фіксовану кількість ресурсів. Кожний ФС може містити певну кількість ВМ.

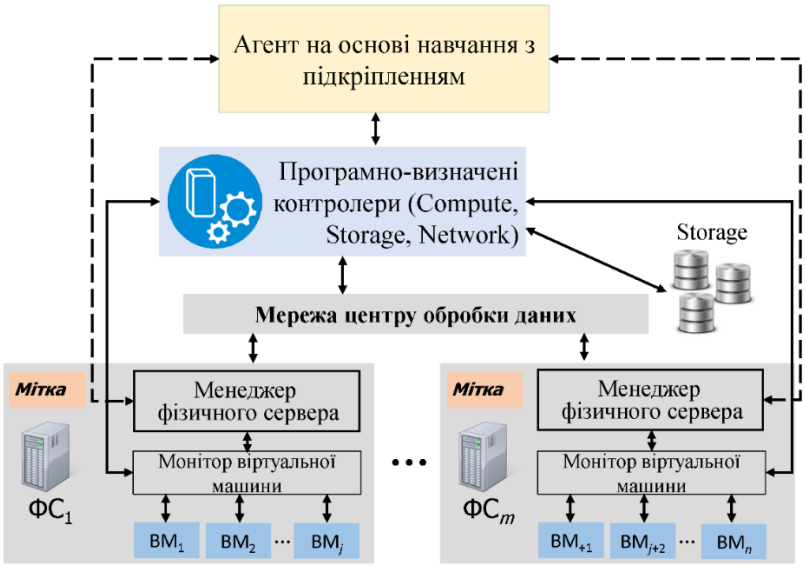


Рис. 5.1. Схема роботи агента на основі навчання з підкріпленням

ЦОД орієнтований на обробку пакетних та транзакційних завдань, які вирішуються в рамках трьох основних сервісних моделей IaaS, PaaS та SaaS. Модель IaaS є основною для обслуговування сучасних ЦОД. Зберігання даних забезпечується централізованим сховищем даних.

Забезпечення ресурсами здійснюється на основі їх наявності для уникнення надмірного виділення ресурсів ЦОД. При цьому зміни клієнтських вимог до ресурсів можуть збільшуватися до певного рівня не впливаючи на інших клієнтів та у відповідності до SLA. Сучасна практика надмірного виділення (резервування) ресурсів з метою попередження небажаних наслідків, спричинених браком ресурсів, із збільшенням навантаження та потреб клієнтів, призводить до збільшення споживання енергії та експлуатаційних витрат.

5.5.3. Агент навчання з підкріпленням в системі управління центром оброблення даних

Під час процесу управління ресурсами агент навчання наближається до найкращої політики управління через перегляд існуючої політики управління у відповідь на зміни стану хмарного ЦОД. Основним недоліком тут є швидкість конвергенції, яка може не задовольняти вимогам SLA або не може мінімізувати споживання енергії хмарним ЦОД. Причиною цього є те, що агенту навчання необхідно виконати певну кількість попередніх дій, що впливають на нові стани

середовища. Ще одним недоліком є великий розмір простору станів. Експериментальні дослідження роботи запропонованого методу проведено для зменшеного розміру простору станів в порівнянні з [324], зменшуючи кількість інтервалів використання ресурсів і зменшуючи кількість атрибутів простору станів.

Послідовність операцій взаємодії компонентів СУІ при динамічному розміщенні і консолідації ВМ (рис. 5.2):

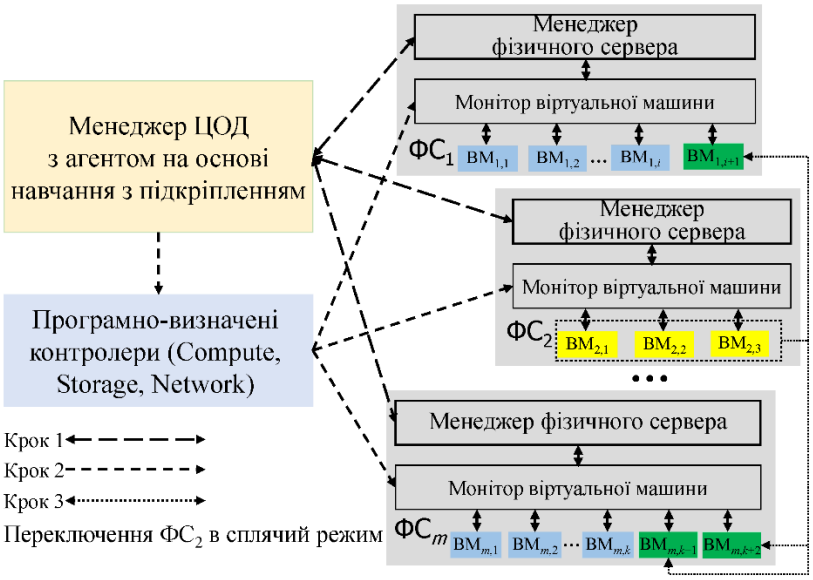


Рис. 5.2. Схема взаємодії компонентів СУІ при динамічному розміщенні ВМ

1) спостереження агентом навчання за поточним станом ФС. Агент НП отримує інформацію про поточні стани ФС від МФС, після надходження завдань або запитів на розміщення ВМ від користувачів. Ця інформація являє собою поточне використання ресурсів ФС. Після цього агент обирає режим роботи для кожного ФС на основі алгоритму *Q*-learning. МФС обчислює наявну ресурсну ємність ФС, таку як центральний процесор (ЦП), пам'ять (RAM), пропускну здатність мережі (BW) та їх прогнозоване використання;

2) надсилання керуючих впливів до МВМ. Оптимізація розміщення ВМ відбувається залежно від рішення агента щодо режиму роботи ФС. Якщо агент обрав сплячий режим, то всі ВМ на ФС повинні мігрувати до інших ФС. Вибір ВМ для міграції з перевантаженої ФС відбувається відповідно до політики вибору. Ця політика полягає у виборі ВМ, яка вимагає мінімального часу міграції, ніж інші ВМ на ФС. Час міграції розраховується через ділення обсягу

пам'яті, призначеної для ВМ, на доступну пропускну здатність мережі між поточним і цільовим ФС. Після вибору ВМ відбувається пошук цільового ФС;

3) команди міграції ВМ. МФС надсилають команди міграції до ВМ, які повинні мігрувати на інші ФС.

5.5.4. Модель агента навчання з підкріпленням в системі управління IT-інфраструктурою

Ефективний метод розміщення повинен зменшити кількість активних ФС відповідно до поточного навантаження на ФС ЦОД. Потрібно приймати рішення про те, коли перевести ФС в сплячий або активний режим. З цієї причини пропонується агент НП як важлива частина алгоритму динамічного розміщення ВМ. Агент обирає режим ФС на основі попередніх даних і отримує винагороду/штраф при зміні стану середовища. З цією метою агент вивчає політику виявлення режиму ФС на вхідних запитах та налаштовує відповідну політику за допомогою *Q*-learning.

В момент часу *t* агент сприймає поточний стан усіх ФС, виконує пошук дії за допомогою алгоритму *Q*-learning та відправляє знайдену дію на виконання до МФС (рис. 5.2). Кожен елемент з простору станів *S* представляє поточну обчислювальну потужність процесорів, об’єм оперативної пам'яті, пропускну здатність мережі та показник запису/читання жорсткого диску усіх ВМ на кожному ФС:

$$\begin{aligned} s_t = \{ & \{CPU_{PM1}, RAM_{PM1}, NET_{PM1}, STIO_{PM1}\}, \\ & \{CPU_{PM2}, RAM_{PM2}, NET_{PM2}, STIO_{PM2}\}, ..., \\ & \{CPU_{PMi}, RAM_{PMi}, NET_{PMi}, STIO_{PMi}\}, \\ & \{CPU_{PMm}, RAM_{PMm}, NET_{PMm}, STIO_{PMm}\}\} \end{aligned}, \quad (5.9)$$

де $i = \overline{1,m}$, m – кількість ФС, $CPU \in [0,1]$ – використана обчислювальна потужність процесора, $RAM \in [0,1]$ – використаний об’єм оперативної пам'яті, $NET \in [0,1]$ – показник використаної завантаженості мережі, $STIO \in [0,1]$ – показник використання запису/читання жорсткого диску. Кожен показник

використання ресурсу нормалізується відносно максимального об'єму відповідного ресурсу ФС.

Кількість доступних ФС (значення q в

Алгоритм 5.4.) вибирається залежно від поточної стратегії управління, яка враховує динаміку створення ВМ в ЦОД. Динаміка створення ВМ на поточний момент часу може бути оцінена за допомогою середнього коефіцієнту життєздатності ВМ (3.23). Метрика V_{VM}^{mid} запропонована автором в статті [322].

Метрика V_{VM}^{mid} використовується на різних етапах прийняття рішень щодо управління ВМ та ФС. Якщо $V_{VM}^{mid} < 1$ на короткому проміжку часу, то стратегію управління потрібно скоригувати при планування таким чином, щоб запустити процес консолідації більш «наполегливо» і значення q буде наближено до кількості увімкнутих ФС. Якщо $V_{VM}^{mid} > 1$ для короткого проміжку часу, то відповідна стратегія управління повинна увімкнути деяку кількість ФС і значення q буде меншим (або набагато меншим), чим кількість увімкнутих ФС.

Ефективність процесу консолідації на заданому відрізку часу пропонується оцінювати за допомогою метрик: кількість порушень SLA, кількість увімкнутих ФС, кількість міграцій ВМ. Агент НП вирішує, коли перемикати РМ в сплячий режим або в активний режим.

На першій стадії експериментальних досліджень показник використання кожного ресурсу був розбитий на чотири інтервали:

$$\begin{aligned} &CPU \in \{[0,0.25),[0.25,0.5),[0.5,0.75),[0.75,1]\}, \\ &RAM \in \{[0,0.25),[0.25,0.5),[0.5,0.75),[0.75,1]\}, \\ &NET \in \{[0,0.25),[0.25,0.5),[0.5,0.75),[0.75,1]\}, \\ &STIO \in \{[0,0.25),[0.25,0.5),[0.5,0.75),[0.75,1]\}. \end{aligned} \quad (5.10)$$

Тобто, зміна показників в межах інтервалів не призведе до переходу в інший стан.

Агент виконує дію на основі спостережуваного стану середовища. Простір дій визначається як набір $A_t = \{a_{t1}, a_{t2}, ..., a_{ti}, a_{tm}\}$, де $a_{ti} \in \{-1, 0, 1\}$, $i = \overline{1, m}$. Кожна дія з A переводить ФС у сплячий або активний режим роботи до наступного інтервалу часу $t+1$. Алгоритм динамічного розміщення ВМ переводить кожну

ФС у режим роботи на основі рішення агента НП. Потім агент отримує винагороду/штраф та вираховує *Q*-значення після зміни всіх режимів роботи до початку інтервалу часу $t+1$, але після завершення виконання обраної дії. Основною метою алгоритму динамічного розміщення ВМ є мінімізація споживання електроенергії та кількості порушень SLA, обчислюючи значення штрафу p_t, яке складається з двох значень: штраф за порушення SLA p_t^{SLA} та штраф за споживання електроенергії p_t^{power}.

$$p_t = \beta p_t^{SLA} + \delta p_t^{power}, \tag{5.11}$$

де α та β – ваги, що визначають відносну важливість P_t^{SLA} та P_t^{power} відповідно.

Штраф за порушення SLA. Досягнення бажаних вимог QoS є надзвичайно важливим для середовища хмарних обчислень. Вимоги QoS зазвичай визначаються в термінах SLA, які описують такі характеристики, як пропускна спроможність або час відгуку. Оскільки ці характеристики можуть змінюватися для різних застосунків, необхідно визначити метрику, яка не залежить від робочого навантаження і може бути використана для оцінки SLA будь-якої ВМ, що розгортається в ЦОД.

Якщо фактичне використання *k*-го ресурсу *j*-ою ВМ на *i*-ій ФС $res_{ija}^k(t) \in [0,1]$ більше ніж очікуване використання *k*-го ресурсу $res_{ije}^k(t) \in [0,1]$ за інтервал часу, то це вважається порушенням SLA:

$$res_{ija}^k(t) > res_{ije}^k(t).$$

Штраф за порушення SLA розраховується через ділення сумарної кількості порушень SLA по усім ФС після виконання дії $a_t \in A$ на сумарну кількість порушень на попередньому кроці управління $a_{t-1} \in A$:

$$p_t^{SLA} = \begin{cases} \sum_{i=1}^{m} \frac{T_t^{SLA}}{T_{t-1}^{SLA}}, & T_{t-1}^{SLA} > 0 \\ 0, & T_{t-1}^{SLA} = 0 \end{cases}, \tag{5.12}$$

де T_t^{SLA} – час порушень SLA після виконання дії $a_t \in A$; T_{t-1}^{SLA} – час порушень SLA після завершення виконання дії $a_{t-1} \in A$.

Якщо $P_t^{SLA} < 1$ то це означає, що агент вибрав правильну дію, щоб мінімізувати кількість порушень SLA.

Штраф за збільшення споживання електроенергії. Штраф розраховується як загальна сума відношень споживання електроенергії усіх ФС на попередньому і поточному кроках:

$$p_t^{power} = \sum_{i=1}^{m} \frac{P_{i,t}}{P_{i,t-1}}, \tag{5.13}$$

де $P_{i,t}$ – значення споживання електроенергії в поточному кроці; $P_{i,t-1}$ – значення споживання електроенергії на попередньому кроці управління.

Усі розміщення та розподіли ВМ, а також зміна режиму роботи ФС, повинні завершитися до початку наступного кроку управління t+1. Тоді Q-значення для кожної пари «стан-дія» часового інтервалу оновлюється через загальну суму штрафів P_t.

Q-значення пари стан-дія $Q(s_t, a_t)$ являє собою очікувані сумарні витрати електроенергії та кількості порушень SLA, спричинені дією, прийнятою у стані $s_t \in S$. Коли агент наступного разу спостерігає за станом, він вибирає режим роботи ФС, який забезпечує мінімальне Q-значення. Агент НП вибере дію, що має найменше Q-значення (кількість порушень SLA і витрати електроенергії). Алгоритм роботи агента НП на інтервалі управління від t до t+1 має такі етапи:

Крок 1. Для кожної пари $s \in S$ та $a \in A$, ініціалізувати Q-значення, що дорівнює нескінченності (виконується один раз на початку роботи алгоритму);

Крок 2. Отримати поточний стан $s_t \in S$;

Крок 3. Обрати дію $a_t \in A$ на основі статичного порогу або за допомогою рівняння (5.8);

Крок 4. Виконати дію $a_t \in A$;

Крок 5. Обчислити штраф за порушення SLA p_t^{SLA} за допомогою рівняння ((5.12);

Крок 6. Обчислити штраф за збільшення споживання електроенергії p_t^{power} за допомогою рівняння (5.13);

Крок 7. Обчислити сумарний штраф P_t після зміни режиму роботи, використовуючи рівняння (5.11);

Крок 8. Отримати новий стан $s_{t+1} \in S$ та оновити значення $Q(s_t, a_t)$ за допомогою рівняння (5.7).

Крок 9. Збереження нового стану $s \leftarrow s_{t+1}$;

Крок 10. Повернутися до кроку 2.

Під час початку процесу навчання та кожного разу, коли агент не відвідував поточний стан раніше, пропонується застосовувати дії, що враховують нижнє або верхнє порогові значення використання ресурсів, замість вибору Q-значеннь, що дорівнюють нескінченності. Поріг є більш ефективним ніж випадковий вибір у звичайному Q-learning. Якщо використання ресурсів ФС не перевищує, наприклад, 40% від загальної кількості доступних ресурсів, то агент переводить ФС в сплячий режим роботи. Також, на ранніх стадіях навчання агента, перспективним є використання підходу НП одночасно з іншими політиками управління, наприклад [207].

5.5.5. Метод динамічної консолідації і розміщення віртуальних машин на основі навчання з підкріпленням

В дисертації запропоновано метод динамічної консолідації і розміщення ВМ на основі НП з метою зменшення витрат електроенергії в ЦОД та зменшення кількості порушень SLA. SLA включає в себе вимоги та обмеження щодо якості послуг, що надаються провайдером: часу відгуку, доступності, пропускної здатності, безпеки та ін. SLA виконується для кожного клієнта, коли вся продуктивність, яку потребують застосунки всередині ВМ, забезпечується в будь-який час.

ВМ розміщуються на мінімальній кількості ФС відповідно до поточних затребуваних ресурсів. Коли використання ресурсів ФС є низьким, всі ВМ перерозподіляються на інші ФС, а недостатньо завантажений ФС переходить до сплячого режиму. Крім того, коли ФС перевантажена, деякі ВМ повинні бути переміщені, щоб зменшити кількість порушень SLA. Якість алгоритму

розміщення ВМ може поліпшуватися у процесі роботи агента НП та алгоритму *Q*-learning для визначення режиму роботи кожного ФС (сплячий або активний).

Алгоритм може динамічно адаптувати ФС до зміни робочого навантаження. Важливою частиною алгоритму є вирішення питання про те, чи (1) додатковий ФС повинен забезпечити ефективне використання ресурсів зі збільшенням робочого навантаження, або (2) резервні ФС можуть бути переведені в сплячий режим або (3) чи достатня поточна кількість ФС. Агент НП вважається важливою частиною алгоритму для прийняття такого рішення.

ФС, які задіяні у процесі оптимізації, позначаються такими мітками:

- HIBERNATE – позначаються ФС, які потрібно перевести до сплячого режиму;
- PLACEMENT – позначаються ФС, які задіяні у процесі розміщення ВМ;
- AVAILABLE – позначаються ФС, які доступні для процесу розміщення ВМ.

Алгоритм динамічного розміщення ВМ на основі НП наведено на рис. 5.3. Результатом роботи алгоритму є список ВМ L^{VM} , які потрібно розмістити в ЦОД. Розподіл та міграції ВМ ілюструє

Алгоритм 5.4. Створюється список L^{PM}, який містить ФС з міткою AVAILABLE. До кожного ФС із списку L^{PM} надсилаються вимоги до обчислювальної потужності процесора, об’єму оперативної пам'яті, пропускної здатності мережі та показника запису/читання жорсткого диску вибраної ВМ зі списку L^{VM}. Якщо ФС має ресурси для ВМ, то такий ФС позначається міткою PLACEMENT та повертає прогнозоване навантаження. Якщо жодний ФС не має ресурсів для ВМ, то ВМ повертається у кінець списку L^{VM}. У іншому разі ВМ мігрує до ФС з максимальним навантаженням. Усі інші ФС позначаються міткою AVAILABLE. Якщо ВМ вдруге обирається зі списку L^{VM} і жодний ФС знову не має ресурсів для ВМ, то така ВМ видаляється зі списку L^{VM} та залишається на поточному ФС.

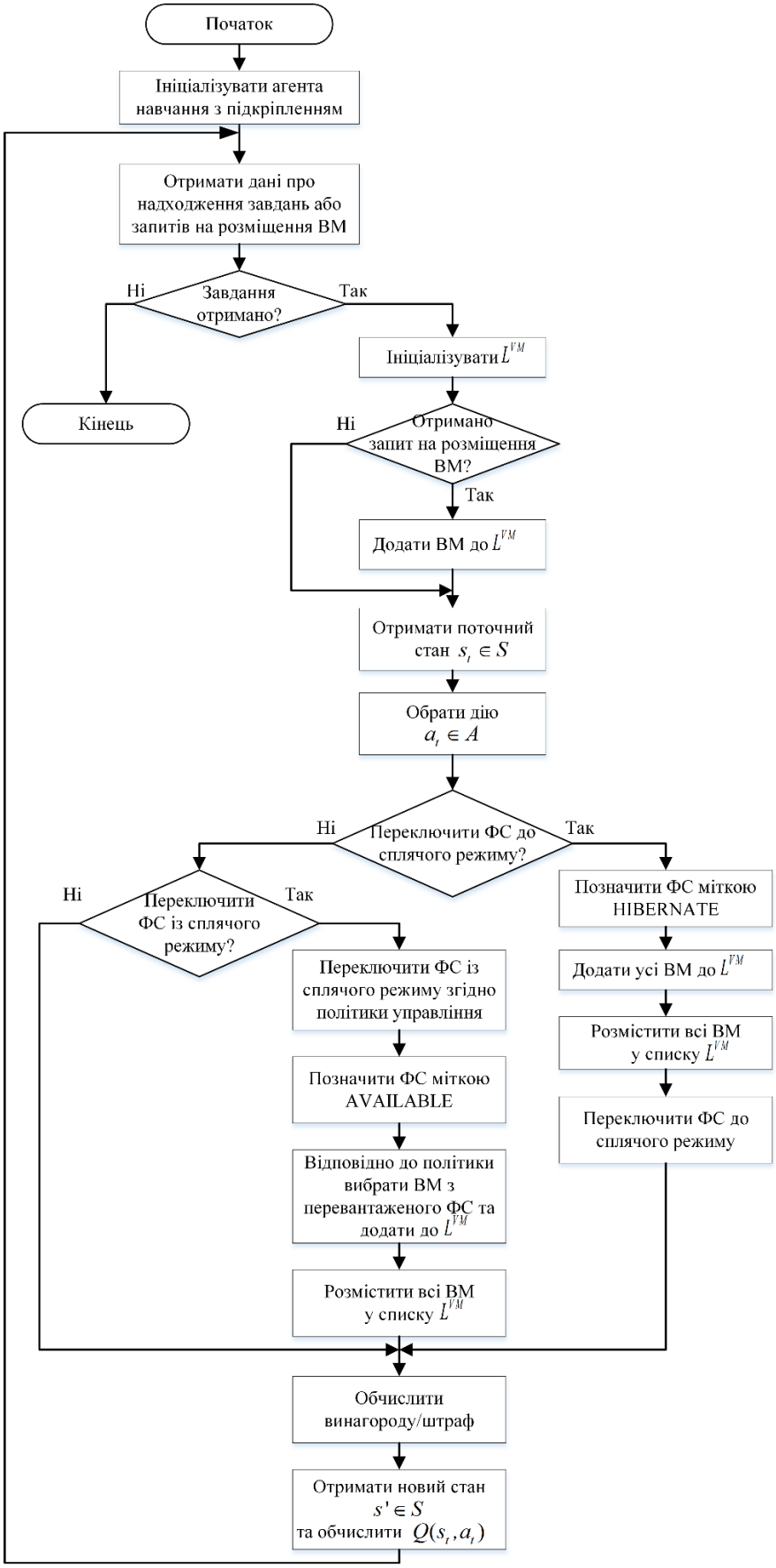


Рис. 5.3. Алгоритм динамічного розміщення ВМ на основі НП

Алгоритм 5.4

Вхід: L^{VM}

Вихід: рішення про розміщення ВМ на ФС

1. **while** L^{VM} is not empty **do**
2. Вибрати ВМ з L^{VM};
3. Ініціалізувати L^{PM};
4. Вибрати q ФС з міткою AVAILABLE та додати їх до L^{PM};
5. $LT^{PM} \leftarrow NULL$;
6. $res_{VM} \leftarrow \{CPU_{VM}, RAM_{VM}, NET_{VM}, STIO_{VM}\}$;
7. **for** $i = 1$ **to** q **do**
8. Надіслати вимоги res_{VM} до L_i^{PM};
9. Отримати відповідь від L_i^{PM};
10. **if** L_i^{PM} має ресурси для ВМ **then**
11. Позначити L_i^{PM} міткою PLACEMENT;
12. Отримати прогнозовані значення Res_{PM} від ФС L_i^{PM};
13. $LT^{PM} \leftarrow L_i^{PM}$;
14. **end if**
15. **end for**
16. **if** LT^{PM} порожній **then**
17. **if** ВМ вдруге вибрана зі списку L^{VM} **then**
18. Видалити ВМ зі списку L^{VM};
19. **else**
20. Повернути ВМ в кінець списку L^{VM};
21. **else**
22. Вибрати ФС з LT^{PM}, що підходить для розміщення ВМ;
23. Розмістити ВМ на ФС;
24. Позначити всі інші ФС в LT^{PM} міткою AVAILABLE;
25. **end if**
26. **end while**

Виконується декілька перевірок перед тим, як ФС підтвердить можливість розміщення ВМ. Повинна виконуватись вимога (5.14) для кожного ресурсу:

$$Res_{PM}^{k,AVAIL}(t) > res_{VM}^{k}(t), \tag{5.14}$$

де $Res_{PM}^{k,AVAIL}(t)$ – доступна кількість ресурсів ФС, $res_{VM}^{k}(t)$ – вимоги до кількості ресурсів з боку ВМ.

ФС починає процес розміщення ВМ, якщо виконується вимога (5.15) для кожного ресурсу:

$$Res_{PM}^{k,MAX}(t) > Res_{PM}^{k,P}(t), \tag{5.15}$$

де $Res_{PM}^{k,MAX}(t)$ – максимальна кількість ресурсів ФС, $Res_{PM}^{k,P}(t)$ – прогнозоване навантаження.

Рівняння (5.16) визначає поточну ємність ЦОД, яка використовується для визначення динаміки використання ресурсів:

$$CAP^{k,DC}(t) = \sum_{j=1}^{n} res_j^k(t) / \sum_{i=1}^{m} Res_i^k(t), \quad (5.16)$$

де n – кількість ВМ в певний момент часу t, m – кількість ФС, $res_j^k(t)$ – вимоги до кількості k-го ресурсу j-ї ВМ, $Res_i^k(t)$ – поточне використання k-го ресурсу i-го ФС.

Реалізація методу динамічної консолідації і розміщення віртуальних машин на основі алгоритму навчання з підкріпленням у вигляді діаграм класів застосунку моделювання наведена у додатку А.

5.6. Метод управління міграцією та розміщенням віртуальних машин в сталому режимі з використанням прогнозування

З усього спектра завдань при управління IT-інфраструктурою хмарного ЦОД особливий інтерес викликає проблеми управління ЦОД в цілому та проблеми управління IT-послугами та відповідними IT-ресурсами. У дисертаційній роботі особлива увага приділяється вирішенню проблеми управління ресурсами з метою забезпечення заданої якості реалізації інформаційних процесів [275]. Проблеми управління IT-інфраструктурою ЦОД в цілому вирішуються за допомогою систем DCIM [262, 263, 264]. При цьому застосовуються бази знань підприємства [343, 344, 345]. Попередні експериментальні дослідження проводилися для вирішення завдання розподілу ВМ на ФС в режимі оф-лайн за допомогою трьох алгоритмів: керованого генетичного алгоритму (УГА) [212], АГА [250] і класичного генетичного алгоритму (КГА).

Основним завданням при реалізації управління хмарним ЦОД є ефективний перерозподіл ресурсів між користувачами послуг в умовах аварій, несправностей, непередбачуваних змін навантаження, збільшення кількості

клієнтів і кількості їх запитів. Перерозподіл ресурсів необхідно здійснювати таким чином, щоб не було випадків порушення SLA при мінімізації споживання електроенергії.

Для вирішення завдання управління ресурсами IT-інфраструктури в режимі онлайн пропонується використовувати адаптивний програмно-визначений підхід з комбінацією централізованого та децентралізованого управління. Використання такого підходу до управління ЦОД дозволяє в умовах програмно-визначеного середовища вибирати стратегії управління ресурсами і ВМ таким чином, щоб здійснювалася адаптація до впливу таких факторів, як зміна навантаження, установка додаткових оновлень, розгортання нового програмного забезпечення і апаратних платформ, внесення змін в структуру і продуктивність послуг, що надаються.

5.6.1. Глобальний менеджер на підставі адаптивного програмно-визначеного управління

У запропонованій СУІ [16] глобальний менеджер ЦОД здійснює централізоване управління хмарним ЦОД через управління фізичними і віртуальними ресурсами і вибором політик управління для адаптації до зовнішніх чинників як показано в п. 2.3. ГМ підтримує властивості адаптивності і програмної визначеності та реалізує метод управління ресурсами ЦОД в сталому режимі.

Адаптивний програмно-визначений підхід полягає у виборі стратегії управління залежно від поточного і передбаченого стану ВМ, серверів і ЦОД в цілому з урахуванням трендів використання ресурсів. ГМ, що реалізує цей підхід називається адаптивним програмно-визначеним менеджером (АПВМ). АПВМ може взаємодіяти з аналогічними по функціональності модулями управління інших ЦОД, обмінюючись інформацією, що управляє по глобальній мережі або виділених каналах зв'язку. АПВМ також реалізує політику допуску клієнтів до сервісів. АПВМ реалізується на базі відмовостійкого кластера з високою доступністю.

АПВМ здійснює управління ресурсами усіх ФС (рис. 5.4). Для цього кожен ФС з'єднаний з АПВМ за допомогою віртуального каналу, створеного на основі політик контролера програмно-визначеної мережі. АПВМ періодично отримує дані про поточний і прогнозований стани ресурсів через отримання повідомлень від кожного сервера. Повідомлення містять мітку, час відправлення, загальну ємність ресурсів ФС (*Res*, для процесора, пам'яті, мережевого та дискового простору) і прогнозовану утилізацію цих ресурсів, обчислену на стороні ФС.

Функції управління ресурсами окремого сервера виконує МФС. Причому *Res* записується в такий спосіб:

$$Res = \{CPU, RAM, NET, STIO\}. \quad (5.17)$$

Для позначення серверів, які задіяні в процесах управління, використовуються мітки. Міткою "CONSOLIDATION" відзначаються сервери, які у цей час знаходяться у процесі консолідації. Міткою "PLACEMENT" відзначені сервери, що використовуються у цей час для розміщення ВМ. Міткою "SCALING" відзначені сервери, що знаходяться у процесі настройки внутрішніх ресурсів, що надаються локальним ВМ на цьому сервері. Міткою "AVAILABLE" позначаються сервери, доступні для розміщення та консолідації ВМ.

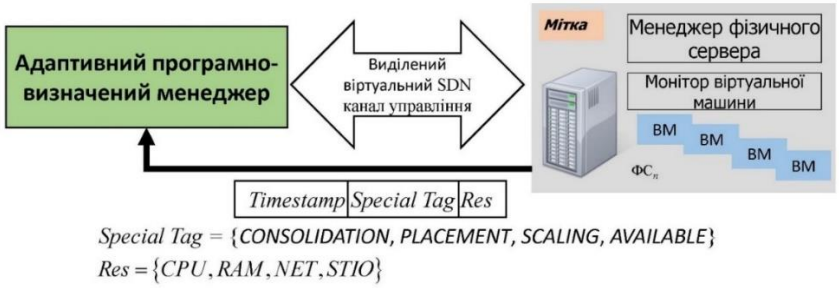


Рис. 5.4. Взаємодія АПВМ і МФС

5.6.2. Функціональність менеджера фізичного сервера

Кожен ФС характеризується використанням ресурсів процесора, пам'яті, мережевого підключення, завантаженням інтерфейсу пристрою зберігання та ін. На кожному сервері виконується МВМ, наприклад Xen [258], ESXi [259] або Hyper-V [260], а також спеціальна ВМ з МФС для управління ВМ засобами МВМ. МФС також може бути модулем в МВМ.

МФС накопичує історичні дані про використання локальних ресурсів сервера. На основі історичних даних МФС обчислює прогноз використання ресурсів на середньостроковий період, який можна порівняти з часом міграції ВМ, що працюють на цьому сервері. Дані про прогноз використання ресурсів разом з міткою часу зберігаються локально і відправляються в АПВМ. МФС генерує команди запуску, зупинки і паузи для ВМ при управлінні локальним МВМ або гіпервізором.

Якщо АПВМ дозволяє децентралізоване управління для груп ВМ, то МФС можуть взаємодіяти один з одним в стійці за допомогою групових повідомлень для вирішення проблем перерозподілу ресурсів всередині стійки за допомогою масштабування ВМ вгору або вшир.

Моніторинг ФС здійснюється безпосередньо за допомогою давачів, системного програмного забезпечення гіпервізора і локальним менеджером МФС на рівні ФС і комутатора стійки. Один з етапів оброблення даних моніторингу є прогнозування потреб в ресурсах для ВМ на ФС. МФС передає прогноз даних про навантаження до АПВМ протягом певного інтервалу часу використовуючи віртуальні канали, сконфігуровані контролером SDN. Результати прогнозування використовуються АПВМ для розрахунку тренду використання ресурсів на рівні ЦОД для вирішення проблем довгострокового планування ресурсів і вибору стратегії управління.

Унаслідок моніторингу МФС виявляє одну або кілька ВМ, для яких не виконуються вимоги SLA. Якщо така ВМ знайдена, МФС позначає себе тегами "SCALING" або "CONSOLIDATION", залежно від того, чи достатньо локальних ресурсів. На підставі політик, отриманих від АПВМ, МФС приймає рішення про виділення більшого обсягу локальних ресурсів цим ВМ. Управління МФС повністю відповідає принципам програмно-визначеному підходу. Алгоритми логіки і функціонування надходять з АПВМ і визначаються поточною стратегією управління хмарним ЦОД.

5.6.3. Метод розміщення і міграції віртуальних машин в режимі реального часу з використанням прогнозу навантаження

Розроблений алгоритм роботи АПВМ розміщення нових, а також управління функціонуючими ВМ [275] наведено на рис. 5.5. Для управління кожним запитом створюється окремий екземпляр процесу управління, який взаємодіє з МФС і виконує запити на міграцію ВМ та створення нових ВМ. З початку обробляються запити від ФС, на яких, за прогнозом, відбудеться нестача ресурсів (overloaded), потім обробляються запити на створення нових ВМ, останніми обробляються запити від серверів, на яких, враховуючи прогноз, залишилось дуже мало ВМ і їх доцільно мігрувати. Унаслідок роботи алгоритму формується множина ВМ G^{VM} , які необхідно розмістити на ФС ЦОД.

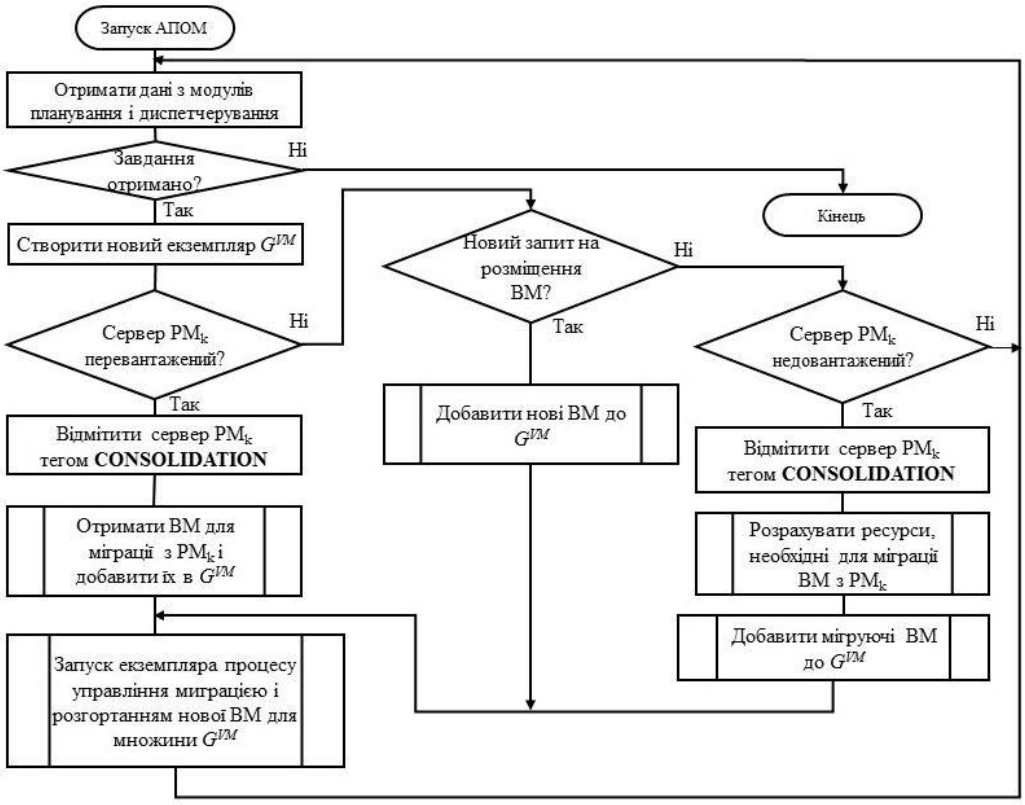


Рис. 5.5. Схема алгоритму управління міграцією та розміщенням ВМ

Алгоритм роботи екземпляра процесу управління, розроблений у працях [275, 308], показаний в алгоритм 5.5. У алгоритмі використані такі позначення: *PM* – ФС; *VM* – ВМ; *N* – кількість ФС, придатних для розміщення ВМ; *K* – індекс ВМ; *P* – стратегія/політика управління; G^{PM} – попередня множина ФС, придатних для розміщення ВМ; GT^{PM} – цільова множина ФС, придатних для розміщення ВМ; Res^{VM} – вимоги до ресурсів з боку ВМ; G_K^{VM} – ВМ з множини G^{VM} .

Алгоритм 5.5. Міграція та розміщення нових ВМ

Input: G^{VM}, *N*, P
Output: Рішення про міграцію або розміщення *VM* на *PM*

1 $K \leftarrow 0$
2 $VM \leftarrow G_K^{VM}$
3 **while** *VM* is not NULL **do**
4 Створити новий екземпляр G^{PM}
5 Використовуючи політику P, отримати *N* придатних серверів для розміщення *VM* і добавити їх в G^{PM}
6 $GT^{PM} \leftarrow$ NULL
7 $Res^{VM} \leftarrow G_K^{VM}\left\{CPU_K^{VM}, RAM_K^{VM}, NET_K^{VM}, STIO_K^{VM}\right\}$
8 **for** *i*=1 to N **do**
9 Відправити вимоги Res^{VM} машини *VM* в PM_i
10 Отримати відповідь від PM_i
11 **if** (PM_i має ресурси для *VM*) and (PM_i відмічений як AVAILABLE) **then**
12 Відмітити PM_i міткою PLACEMENT
13 Отримати значення Res^{PM} від PM_i
14 $GT^{PM} \leftarrow PM_i$
15 **end if**
16 **end for**
17 **if** GT^{PM} is NULL **then**
18 **if** idle PMs exists **then**
19 Використовуючи політику P, включити придатний сервер PM і відмітити його міткою AVAILABLE
20 Розмістити *VM* на *PM*
21 **else**
22 Відправити вимоги Res^{VM} машини *VM* в АПВМ іншого ЦОД *DC*
23 Отримати відповідь від АПВМ іншого ЦОД *DC*
24 **if** response is negative **then**
25 **return** неможливо розмістити *VM*
26 **else**
27 Розмістити всі ВМ з G^{VM} в іншому ЦОД *DC*
28 **return**
29 **end if**
30 **end if**
31 **else**
32 Використовуючи політику P, отримати *PM* з GT^{PM}, придатний для розміщення *VM*
33 Розмістити *VM* на *PM*
34 Відмітити всі PM в GT^{PM}, крім PM, тегом AVAILABLE
35 **end if**
36 $K \leftarrow K+1$
37 $VM \leftarrow G_K^{VM}$
38 **end while**

Алгоритм 5.5 ілюструє роботу екземпляра процесу управління міграцією і розміщенням нових ВМ в режимі онлайн. При обробці отриманого списку G^{VM} АПВМ вибирає схожі ФС з усіх працюючих в даний момент в ЦОД через оброблення локальних даних про прогнозований стан ФС. У результаті, АПВМ отримує множину G^{PM} ФС, які можуть претендувати на розміщення ВМ. Потім АПВМ розсилає повідомлення, що містять вимоги $CPU_K^{VM}, RAM_K^{VM}, NET_K^{VM}, STIO_K^{VM}$ до нової/мігруючої ВМ, ФС з множини G^{PM}. ФС, який має, згідно з прогнозом, достатньо ресурсів для розміщення цієї ВМ, позначає себе міткою "PLACEMENT" і відправляє повідомлення в АПВМ, у якому вказує прогнозований обсяг доступних ресурсів. Після оброблення відповідей, отриманих від ФС, АПВМ створює цільовий список серверів GT^{PM} і реалізує стратегію управління, що базується на адаптивному програмно-визначеному підході. При цьому зі списку GT^{PM} вибирається ФС, найбільш придатний для розміщення заданої ВМ відповідно до обраної в даний момент стратегії, наприклад: вибір ФС з максимальним завантаженням або вибір ФС з таким завантаженням ресурсів, при якому при розміщенні нової ВМ буде мінімальне розбалансування ресурсів (відсутність істотних перекосів в сторону використання одного з ресурсів ФС). Крім того, на повідомлення АПВМ можуть відповідати ФС з достатньою прогнозованою кількістю ресурсів і помічені міткою "AVAILABLE".

ФС з завантаженням процесора менше встановленого порогу не відповідає на повідомлення від АПВМ, оскільки згідно з прогнозом, ВМ що працюють на ньому слід перемістити на інші ФС для переведення цього ФС в сплячий режим, або його виключення, якщо визначений тренд на зменшення навантаження.

Якщо відповіді на повідомлення АПВМ не отримані і відповідний для розміщення ВМ з множини G^{VM} ФС не знайдений, то АПВМ приймає рішення перевести з сплячого режиму або включити ФС в стійці, де знаходиться максимальна кількість працюючих ФС. Потім АПВМ відправляє включеному ФС повідомлення про створення нової ВМ або розміщення у себе мігруючої ВМ і позначає цей ФС міткою "AVAILABLE".

АПВМ приймає рішення, виходячи з аналізу таких даних: значення індикаторів і метрик, зазначених в SLA; завантаження мережевого інтерфейсу ФС і комутатора стійки; чутливості до перевантажень, що створюється процесом міграції; взаємного впливу ВМ однією на одну в межах одного ФС; мінімізації розміщення декількох ВМ з пульсуючими навантаженнями на одному ФС. АПВМ приймає рішення з управління на основі показників SLA, навантаження на ФС, навантаження на комутатор стійки, чутливості до накладних витрат, спричинених міграціями і взаємним впливом ВМ [265].

При цьому АПВМ враховує інформацію про прогноз навантаження на нову ВМ або не враховувати її. Облік такої інформації дозволяє зменшити число зайвих міграцій ВМ в майбутньому.

АПВМ аналізує виконання умови

$$Res_{AVAIL}^{PM} \geq Res_{K}^{VM}, \tag{5.18}$$

де Res_{AVAIL}^{PM} – доступне кількість ресурсів ФС, Res_{K}^{VM} – ресурси, які потрібні для ВМ зі списку G^{VM}.

Якщо умова (5.18) для кожного ресурсу з множини (5.17) задовольняється для деякої множини ФС, то АПВМ обробляє цю множину.

Коли екземпляр процесу управління з вибору цільового ФС закінчений, АПВМ відправляє повідомлення цільовому ФС створити нову ВМ і позначає цей сервер міткою "AVAILABLE". Потім відправляє повідомлення ФС зі списку GT^{PM} для позначки їх як "AVAILABLE". На стороні ФС локальним МФС проводяться кілька перевірок перед тим, як ФС прийме пропозицію про розміщення ВМ.

Здійснюється обчислення

$$Res_{AVAIL}^{PM} = Res_{Th}^{PM} - Res_{P}^{PM}, \tag{5.19}$$

де Res_{Th}^{PM} – максимальний поріг використання ресурсу, Res_{P}^{PM} – прогнозоване використання ресурсу.

Вираз (5.19) для кожного ресурсу з множини (5.17) обчислюється локально менеджером ФС. Якщо виконується одна з умов $CPU_{AVAIL}^{PM} < 0$, $RAM_{AVAIL}^{PM} < 0$,

$NET_{AVAIL}^{PM} < 0$, $STIO_{AVAIL}^{PM} < 0$, то ФС починає процес локальної консолідації. Причому, процес консолідації ВМ серед ФС із завантаженням вище допустимої запускається з найвищим пріоритетом. ФС, кілька ВМ якого будуть мігрувати, позначається міткою "CONSOLIDATION". Таким чином, для розміщення ВМ з G^{VM}, множина ФС GT^{PM} буде містити ФС з міткою "AVAILABLE". При розміщенні нової ВМ визначається, чи входить вона до складу сервісу з масштабуванням і балансуванням навантаження або до складу групи сервісів. Якщо так, то ВМ повинна бути розміщена в тій же стійці або суміжними стійками, щоб забезпечити мінімальний шлях між новою ВМ, що розміщується, і ВМ, що входять до робочого процесу/сервісу.

АПВМ періодично отримує повідомлення про прогноз доступних ресурсів *Res* від всіх ФС в ЦОД через обмін повідомленнями з локальними МФС. Для вибору відповідної стратегії управління використовується метрика ємності ЦОД для кожного ресурсу з (5.17)

$$CAP_{Res}^{DC} = \sum_{i=1}^{N} Res_i^{VM} \; / \sum_{j=1}^{M} Res_j^{PM}, \qquad (5.20)$$

де N – кількість ВМ, що працюють в певний проміжок часу, M – кількість ФС, Res_i^{VM} – запитана кількість ресурсів з (5.17) i-ю ВМ, Res_j^{PM} – поточне використання ресурсів (5.17) j-м ФС. Вираз (5.20) визначає поточну ємність ресурсів ЦОД з метою визначення тренду використання кожного ресурсу.

Прогнозування навантаження здійснюється за допомогою запропонованого у розділі 4 адаптивного методу комбінованого прогнозування навантаження. Запропоновані методи і моделі прогнозування можливо використовувати як для прогнозування завантаженості окремої ВМ з подальшим узагальненням по всім ВМ на одному ФС, так і для прогнозування завантаженості окремого ФС. У останньому випадку на прогнозоване значення може впливати процес міграції ВМ з/до ФС. Базуючись на прогнозованих значеннях МФС може приймати рішення про виділення більшого об’єму ресурсів в межах поточного ФС, або міграції в межах однієї стійки. АПВМ може враховувати прогнозовані значення

завантаженості для розміщення нових ВМ та міграції існуючих в межах стійки, кластеру, поточного або іншого ЦОД.

Ще одна функція МФС полягає у визначенні недовантажених і перевантажених ВМ та їх динамічних характеристик з метою мінімізувати розміщення декількох ВМ з пульсуючим навантаженням одночасно на одному ФС. Таке рішення приймає АПВМ на основі метрик і визначеної стратегії. При цьому виникає задача визначення часових інтервалів щоб отримати необхідні метрики для прийняття рішень з управління ресурсами в режимах, наближених до оптимальних.

5.7. Метод управління потужністю хмарного центру оброблення даних

Розроблений в дисертації метод управління потужністю хмарного ЦОД реалізує стратегії S_4 та S_6 і дозволяє обчислити кількість ФС для обслуговування прогнозованого навантаження у вигляді ВМ. У статті автора [284] пропонується метод обчислення кількості ФС для обслуговування прогнозованого навантаження у вигляді ВМ. Запропонований метод проілюстровано на прикладі застосування тільки до одного ресурсу k. Позначимо змінною D^k суму вимог до k-го ресурсу, що отримана від модуля прогнозування споживання ресурсу. Позначимо вільну ємність *k*-го ресурсу в ЦОД як F^k. Позначимо змінною Δ^k дефіцит (нестачу) ресурсу, що обчислюється за формулою $\Delta^k = D^k - F^k$.

Якщо задовольняється вимога $\Delta^k \le 0$ і є достатньо місця для розміщення максимального запитуваного ресурсу $c^k_{\max,\mathrm{d}}$ на будь-якому працюючому ФС $(c^k_{\max,\mathrm{d}} < C^k_{\mathrm{free}})$, то немає необхідності вмикати додатковий ФС. Якщо $\Delta^k > 0$, то деяка кількість додаткових ФС повинна бути увімкнена. Щоб врахувати гетерогенність усіх ФС, пропонується відсортувати їх типи g у порядку зменшення ємності ресурсу *k*. Передбачається, що чим більше ємність ресурсу *k* у ФС, тим більше він споживає електроенергії. Позначимо змінною $C^{g,k}$ ємність ресурсу k ФС типу g. Наприклад, $C^{1,k} = 1$, якщо ФС типу $g = 1$ має ємність ресурсу *k*, що дорівнює 1 $(k = 1)$, $C^{2,k} = 0{,}75$, якщо ФС типу $g = 2$ має ємність

ресурсу k, що дорівнює 0,75 $(k=0,75)$, і т.д. Позначимо змінною $M_{PM}^{+}(t)$ кількість ФС типу g, які повинні бути увімкнені щоб обслуговувати навантаження з боку нових ВМ без затримки. Позначимо змінною $M_{PM}^{g,off}(t)$ кількість ФС типу g, які знаходяться в режимі сну.

Необхідно врахувати два випадки для виконання подальших обчислень. Перший випадок: якщо $\Delta^k \le 1$, то тільки один ФС з ємністю ресурсу k, що є найближчою до Δ^k (для таких g, коли $\frac{C^{g,k}}{\Delta^k} \ge 1$), повинен бути увімкнений. Другий випадок: якщо $\Delta^k > 1$, то кількість додаткових ФС обчислюються так (5.21):

$$M_{PM}^{+}(t) = \sum_{i=1}^{M}\left(\left\lceil \frac{M_{PM}^{g,off}(t)}{2} \right\rceil \prod_{j=1}^{i} f^{g}(t)\right),$$

$$f^{g}(t) = \begin{cases} 1, & \left\lfloor \frac{\Delta_j^k}{C^{g,k}} \right\rfloor - \left\lceil \frac{M_{PM}^{g,off}(t)}{2} \right\rceil \ge 0 \\ \left\lfloor \frac{\Delta_j^k}{C^{g,k}} \right\rfloor + 1, & \text{інакше} \end{cases}, \qquad (5.21)$$

$$\Delta_{j+1}^{k} = \Delta_j^k - C^{g,k} M_{PM}^{g,off}(t).$$

Таким чином, отримана кількість ФС для увімкнення $M_{PM}^{+}(t)$ складається з $\left\lceil \frac{M_{PM}^{g,off}(t)}{2} \right\rceil$, $g = \overline{1,G}$, ФС кожного типу g. Запропонований метод пропонується використовувати для оцінки нижньої границі кількості ФС, які повинні знаходитися в робочому режимі для розміщення нових ВМ з урахуванням кожного ресурсу k, який враховується при розміщенні ВМ і є критичним для її роботи.

5.8. Модель розподіленого дворівневого сховища з реплікацією

5.8.1. Необхідність управління системою збереження даних в контексті центру оброблення даних

Організація роботи сховища у віртуалізованому середовищі полягає у вирішенні задачі забезпечення заданої продуктивності і відмовостійкості при

доступі до даних з боку ВМ та контейнерів. Один із способів підвищення продуктивності роботи сховища без суттєвого підвищення вартості зберігання даних є розбиття сховища на два рівні: перший рівень є найпродуктивнішим, меншого об’єму і дорожчим при використанні, другий рівень є повільним, більшого об’єму і дешевшим. При організації роботи сховища через розбиття його на рівні виникає необхідність управління процесом міграції даних між рівнями. При цьому, ВМ та контейнери обробляють дані, що знаходяться на пристроях швидкого рівня. У сучасних СЗД швидкий рівень складається з твердотілих накопичувачів. Ці пристрої мають найвищу продуктивність роботи з даними в рамках сховища. Якщо необхідних даних на швидкому рівні немає, відбувається міграція даних з менш швидкісного (повільного) рівня. У системах збереження даних повільний рівень складається з пристроїв на магнітних дисках (HDD). Міграція з повільного рівня на швидкий виконується з метою зменшення затримки при роботі з даними в сховищі. Одночасно з цим відбувається міграція даних зі швидкого рівня на повільний рівень з метою звільнення місця на пристроях швидкого рівня для подальшого оброблення та міграції «гарячих» даних.

Пристрої в складі СЗД можуть виходити з ладу, що може призвести до втрати даних і зупинки роботи сервісів. З метою забезпечення відмовостійкості виникає необхідність управління процесом реплікації даних на інші вузли зберігання залежно від завантаженості вузла операціями вводу/виводу, затримки передачі даних між вузлами та наявності вільного місця. При цьому, репліки даних повинні розташовуватись на вузлах зберігання даних таким чином, щоб забезпечити рівномірне розміщення даних на вузлах і високу продуктивність доступу до даних.

Таким чином, в дисертації розроблено модель розподіленого дворівневого сховища і метод управління, що дозволяє без суттєвого збільшення вартості зберігання даних забезпечити відмовостійкість роботи сховища, а також підвищити продуктивність операцій читання/запису даних ВМ та контейнерами в гіперконвергентних і хмарних ЦОД. Основні результати розроблення і

дослідження моделей і методів управління сховищем представлені у працях [287, 317, 326, 346].

Основними елементами моделі розподіленого дворівневого сховища з реплікацією є дані (файли та блоки), пристрої швидкого рівня, пристрої повільного рівня та ФС. Кожний сервер має локальне сховище згідно схеми підключення DAS. Це сховище включає в себе декілька пристроїв зберігання даних, які розміщені в загальному випадку на декількох рівнях, залежно від їх продуктивності. У розробленій моделі пропонується використовувати два рівні сховища: швидкий і повільний.

Позначимо змінною M кількість ФС в ЦОД, а змінною n_m кількість файлів, які зберігаються ВМ в локальному сховищі поточного ФС. *У* загальному випадку кількість файлів є змінною. Збереження файлів у сховищі відбувається блочно, розмір блоків залежить від умов використання і операційних систем. Позначимо змінною t_m кількість блоків, в яких зберігаються файли m-го сервера. При постановці задачі розмір блока дорівнює 4 Кб. Відповідно, кількість блоків теж змінюється у процесі роботи сховища. Кількість пристроїв швидкого рівня на m-му ФС позначимо як u_m, кількість пристроїв повільного рівня – v_m. Для опису моделі і вирішення задач управління роботою сховища поточного ФС введемо такі змінні:

f_{im} - розмір i-го файлу на сервері m, $i=\overline{1,n_m}$, $m=\overline{1,M}$,

c – розмір блоку даних,

q_{jm} – кількість разів доступу до j-го блоку сервера m на швидкому рівні протягом заданого часу, $j=\overline{1,t_m}$, $m=\overline{1,M}$,

p_{jm} – кількість разів доступу до j-го блоку на повільному рівні протягом заданого часу, $j=\overline{1,t_m}$, $m=\overline{1,M}$,

B – матриця розбиття файлів на блоки (b_{ij}=1 якщо j-ий блок є частиною i-го файлу, b_{ij}=0 – в іншому випадку), $i=\overline{1,n_m}$, $j=\overline{1,t_m}$,

R – матриця розміщення блоків (реплікації) на ФС (r_{jm}=1 якщо j-й блок розміщений на сервері m, r_{jm}=0 – в іншому випадку),

s_{km} – розмір k-го швидкого пристрою на сервері m, $k=\overline{1,u_m}$, $m=\overline{1,M}$,

X – матриця розміщення блоків на швидких пристроях, x_{kj} – логічна змінна розміщення j-го блоку на k-ому швидкому пристрої, $k = \overline{1,u_m}$, $j = \overline{1,t_m}$,

h_{lm} – розмір l-го повільного пристрою на сервері m, $l = \overline{1,v_m}$, $m = \overline{1,M}$

Y – матриця розміщення блоків на повільних пристроях, y_{lj} – логічна змінна розміщення j-го блоку на l-ому повільному пристрої, $l = \overline{1,v_m}$, $j = \overline{1,t_m}$,

Z – вектор значень затримки передачі даних з поточного сервера m на інші сервери ЦОД, що зберігають копії блоків (репліки) сервера m, z_m=0. Індекс вектора відповідає номеру сервера,

E – вектор показників роботи кожного вузла зберігання даних, що зберігає значення вільного місця на повільному рівні для кожного серверу ЦОД, індекс вектора відповідає номеру сервера,

W – вектор показників роботи кожного вузла зберігання даних, що зберігає значення кількості транзакцій доступу до блоків даних на обох рівнях для кожного серверу ЦОД, індекс вектора відповідає номеру сервера,

C_k - вартість використання k-го швидкого пристрою, $k = \overline{1,u_m}$,

G_k - вартість зберігання одного блоку на k-му швидкому пристрої, $k = \overline{1,u_m}$,

D_l - вартість використання l-го повільного пристрою, $l = \overline{1,v_m}$,

g_l - вартість зберігання блоку на l-му повільному пристрої, $l = \overline{1,v_m}$,

d_{vk} – вартість міграції блоку з l-ого повільного пристрою на k-ий швидкий пристрій, $k = \overline{1,u_m}$, $l = \overline{1,v_m}$.

5.8.2. Модель міжрівневої міграції

Робота з блоками відбувається на швидкому рівні. Якщо необхідного блоку на швидкому рівні немає, то відбувається зчитування його з повільного рівня, відправка у відповідні канали обміну з пам’яттю, запис блоку на швидкий рівень та видалення блоку з повільного рівня. Подальша робота з цим блоком відбувається вже читанням/записом зі швидкого рівня. У випадку, коли неможливо записати блок на швидкий рівень, робота з блоком відбувається читанням/записом з повільного рівня до тих пір, поки не з’явиться вільне місце

на швидкому рівні. При міграції блоку зі швидкого рівня на повільний рівень спочатку відбувається його запис на повільний рівень, а потім видалення зі швидкого рівня. Відповідно, при міграції блоку з повільного рівня на швидкий рівень, спочатку виконується запис на швидкий рівень, а потім видалення з повільного рівня. Таким чином, блок існує в одному екземплярі при закінченні процесу міжрівневої міграції.

Причиною для міграції між рівнями постає наявність/відсутність доступу до блоків даних протягом деякого часу. Через це необхідна максимізація середньої кількості транзакцій роботи з блоками, які розміщуються на пристроях швидкого рівня (5.22).

$$\max \sum_{k=1}^{u_m} \frac{\sum_{j=1}^{t_m} q_{jm} x_{kj}}{\sum_{j=1}^{t_m} x_{kj}} \quad (5.22)$$

Кількість блоків на поточному ФС знаходиться з використанням рівняння:

$$t_m = \frac{\sum_{i=1}^{n_m} f_i}{c}.$$

Кількість блоків (без урахування реплік) на всіх вузлах зберігання даних, а саме ФС, знаходиться з використанням такого рівняння:

$$t = \sum_{i=1}^{m} t_i.$$

В моделі треба врахувати такі обмеження:

1. Наявність вільного місця на пристроях швидкого рівня позначимо як

$$\sum_{k=1}^{u_m} \sum_{j=1}^{t_m} x_{kj} c < \sum_{k=1}^{u_m} s_k \quad (5.23)$$

2. Наявність вільного місця на пристроях повільного рівня позначимо як:

$$\sum_{l=1}^{v_m} \sum_{j=1}^{t_m} y_{lj} c < \sum_{l=1}^{v_m} h_l \quad (5.24)$$

3. На пристроях швидкого рівня (5.25) та пристроях повільного рівня (5.26) *j*-й блок повинен зберігатись лише в одному екземплярі:

$$\sum_{k=1}^{u_m} x_{kj} = 1 \quad (5.25)$$

$$\sum_{l=1}^{v_m} y_{lj} = 1. \quad (5.26)$$

4. Кожен з блоків *j* належить тільки одному файлу (5.27):

$$\sum_{i=1}^{n_m} b_{ij} = 1,\ j = \overline{1, t_m}. \quad (5.27)$$

Кількість пристроїв для збереження даних на кожному рівні визначається, з одного боку, можливостями інтерфейсів підключення, а з іншого – вартістю використання пристроїв. Вартість збереження даних на всіх швидких пристроях визначається у такий спосіб (5.28):

$$\sum_{k=1}^{u_m} C_k \quad (5.28)$$

Вартість збереження даних на всіх повільних пристроях визначається за допомогою формули (5.29):

$$\sum_{l=1}^{v_m} D_l \quad (5.29)$$

Вартість зберігання блоків даних на *k*-ому швидкому пристрої знаходиться за допомогою виразу (5.30):

$$\sum_{j=1}^{t_m} G_k x_{kj} \quad (5.30)$$

Вартість зберігання блоків даних на *l*-ому повільному пристрої знаходиться за допомогою виразу (5.31):

$$\sum_{j=1}^{t_m} g_l y_{lj} \quad (5.31)$$

У випадках, коли зберігання даних на пристроях рівня з кращою продуктивністю може стати невиправданим (дорогим), то необхідно мігрувати ці дані на рівень з нижчою продуктивністю, де зберігання даних буде більш дешевим. Вартість міграції між двома пристроями різних рівнів розраховується у такий спосіб (5.32):

$$\sum_{l=1}^{v_m}\sum_{k=1}^{u_m} d_{lk} \,. \qquad (5.32)$$

Таким чином, враховуючи вирази (5.36) – (5.32), мінімізація вартості збереження даних у локальному сховищі ФС з можливістю міграції даних між пристроями різних рівнів виконується за допомогою критерію:

$$\min\left[\sum_{k=1}^{u_m}(C_k+\sum_{j=1}^{t_m}G_k x_{jk})+\sum_{l=1}^{v_m}(D_l+\sum_{j=1}^{t_m}g_l y_{lj})+\sum_{l=1}^{v_m}\sum_{k=1}^{u_m} d_{lk}\right], \qquad (5.33)$$

за умов виконання обмежень (2) – (6).

5.8.3. Модель міжвузлової реплікації

Для забезпечення відмовостійкості при збереженні даних в розподіленому середовищі необхідно вибрати фактор реплікації (replication factor, RF). Фактор реплікації показує, скільки копій даних зберігається в розподіленому сховищі. Зазвичай використовується *RF*=3, що забезпечує такі схеми захисту даних, як висока доступність (high availability, HA) та відмовостійкість (fault tolerance, FT).

Основна задача при створенні нової репліки – обрати вузли зберігання даних, які мають достатню продуктивність дискової і мережевої підсистем для оброблення запитів на запис репліки. У моделі міжвузлової реплікації кожен ФС є вузлом зберігання даних за схемою DAS.

В умовах реального ЦОД намагаються рівномірно завантажити вузли зберігання даних репліками з метою запобігти появі перевантажених вузлів. При цьому можливо використати один з двох підходів: централізований і децентралізований. При централізованому управлінні реплікаціями необхідно виділити спеціальний вузол, який буде зберігати спеціальні метадані, які використовуються для поновлення роботи та регенерації у разі виходу з ладу вузлів зі сховищами даних. При децентралізованому управлінні реплікаціями метадані розміщуються на кожному вузлі мережі сховищ і відновлення роботи системи виконується за рахунок міжвузлової взаємодії.

Далі, розроблення методів і алгоритмів проводиться з урахуванням централізованого сховища метаданих (сервер метаданих).

Сервер метаданих зберігає матрицю **R** та вектори **E**, **W**, елементи яких вираховуються сервером метаданих по мірі надходження даних з вузлів збереження даних. Вектор **E** використовується методом реплікації на кожному вузлі збереження даних m з метою визначення вузлів-реплік з максимальним обсягом вільного місця на повільному рівні. Елементи e_m вектора **E** визначаються за допомогою формули:

$$e_m = \sum_{l=1}^{v_m} h_{lm} + \sum_{k=1}^{u_m} s_{km} - \sum_{j=1}^{t_m} cr_{jm} \, . \tag{5.34}$$

Щоб врахувати кількість транзакцій доступу до блоків на швидкому і повільному рівнях вузлів-реплік використовується вектор **W**. Елементи w_m вектора **W** визначаються за допомогою формули:

$$w_m = \sum_{j=1}^{t_m} (p_{jm} + q_{jm}) r_{jm} \, \square \tag{5.35}$$

Кількість даних, які зберігаються на вузлі, включаючи репліки даних з інших вузлів зберігання даних, не повинні перевищувати загальний розмір накопичувачів даного вузла:

$$\exists m = \overline{1, M} : \sum_{j=1}^{t} cr_{jm} \le \sum_{l=1}^{v_m} h_{lm} \, . \tag{5.36}$$

Кількість копій блоків на вузлах збереження даних повинна задовольняти рівнянню:

$$\exists j = \overline{1, t} : \sum_{m=1}^{M} r_{jm} = RF \, . \tag{5.37}$$

Таким чином, враховуючи вирази (5.34) – (5.37), рівномірний розподіл реплік блоків даних по вузлах зберігання даних досягається мінімізацією стандартного відхилення значень вільного місця на повільному рівні і стандартного відхилення значень кількості транзакцій доступу до блоків даних на кожному сервері ЦОД:

$$\min\left[\sigma_E + \sigma_W\right], \tag{5.38}$$

за умов виконання обмежень (5.23) – (5.27).

5.9. Метод міграції даних між рівнями сховища

Розроблений в дисертації метод управління міжрівневою міграцією даних у сховищі хмарного ЦОД використовується при всіх стратегіях управління. Ідея методу управління дворівневим сховищем ФС базується на міграції даних між швидким та повільним рівнями сховища і полягає у такому. Локальний пул дисків ФС розбивається на два рівні: швидкий і повільний (рис. 5.6).

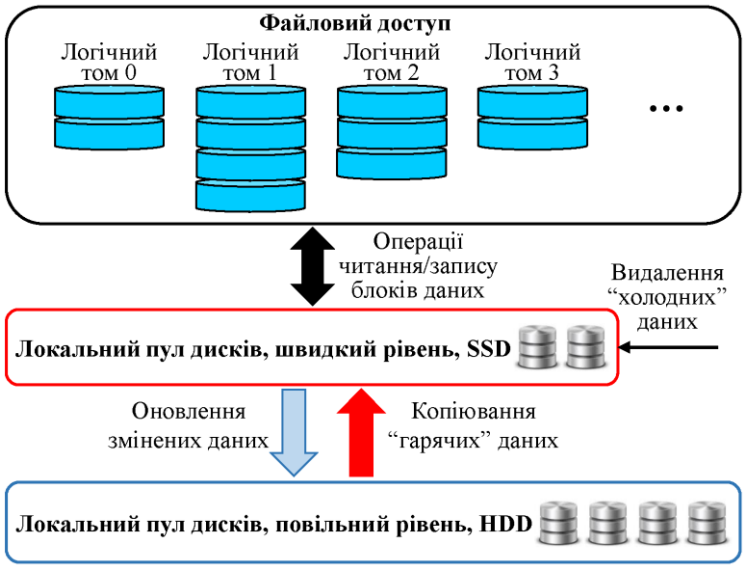


Рис. 5.6. Схема управління міграцією даних дворівневого локального сховища

Застосунки, що виконуються у ВМ, в будь-який момент часу використовують не всі дані логічних томів (дисків). Якщо виникають запити до певних файлів протягом деякого часу необхідно мігрувати відповідні блоки даних з повільного рівня на швидкий. Якщо запити до цих файлів зникають і блоки даних цих файлів більше не потрібні, то виконується їх міграція на повільний рівень, що призводить до вивільнення місця на пристроях швидкого рівня. Таким чином, щоб на рівні швидких пристроїв було в наявності місце, необхідно постійно підтримувати частину вільного місця, яку можливо використовувати для зберігання «гарячих» даних. Якщо створюються нові дані, то в першу чергу вони розміщуються на пристроях швидкого рівня.

В основу метода покладено такі алгоритми: алгоритм сортування файлів за двома критеріями, алгоритм міграції файлів зі швидкого рівня на повільний та алгоритм міграції файлу з повільного рівня на швидкий. Для забезпечення роботи методу управління дворівневим сховищем необхідно сортувати файли за

двома критеріями: за розміром файлу і за кількістю транзакцій доступу до файлу. Кожен з критеріїв може мати певну вагу, яку треба визначати експериментально.

Підготовку списку файлів, які розміщуються на швидкому рівні сховища даних, до міграції на повільний рівень ілюструє Алгоритм 5.6. Робота цього алгоритму базується на сортуванні файлів за збільшенням кількості транзакцій доступу до файлів та за зменшенням їх розміру.

Позначимо як L^F – список файлів швидкого рівня; $migrL^F$ – список файлів швидкого рівня, визначених для міграції на повільний рівень; функція $size(L_i^F)$ повертає розмір i-го файлу.

Алгоритм 5.6 Підготовка списку файлів для міграції на повільний рівень
Вхідні дані: L^F

Вихідні дані: $migrL^F$

1. Ініціалізація списку
   $migrL^F \leftarrow$ NULL
2. **if** $\sum_{k=1}^{u_m} s_k < Th^{SSD}$ **then**
3. $L^F \leftarrow$ сортувати L^F за збільшенням кількості транзакцій доступу до файлів та за зменшенням розміру файлів
4. $wsize \leftarrow size(L_0^F)$
5. $i \leftarrow 1$
6. $migrL^F \leftarrow L_0^F$
7. **while** $wsize < Th^{SSD}$ **do**
8. $migrL^F \leftarrow L_i^F$
9. $wsize \leftarrow wsize + size(L_i^F)$
10. $i \leftarrow i+1$
11. **end while**
12. **return** $migrL^F$

Нові файли завжди за можливістю зберігаються на швидкому рівні сховища. Так як кількість вільного місця на швидкому рівні не повинна бути менше визначеного порогового значення, файли з найменшою кількістю доступів і найбільшого розміру необхідно мігрувати зі швидкого рівня. Процес міграції відбувається паралельно з роботою швидкого рівня по обслуговуванню транзакцій читання/запису. Блоки даних мігрують на повільний рівень поки на швидкому рівні кількість вільного місця не стане більше визначеного порогового

значення Th^{SSD}. Алгоритм 5.6 має складність O($2a \times \log 2a$), де a – кількість файлів в списку L^F.

Значення Th^{SSD} залежить від інтенсивності роботи з файлами великого розміру і підбирається експериментально. Алгоритм враховує, що на пристроях повільного рівня завжди є місце для мігруючих файлів. Алгоритм міграції блоків на повільний рівень (Алгоритм 5.7) працює у такий спосіб:

Алгоритм 5.7. Міграція блоків файлу зі швидкого рівня на повільний рівень
Вхідні дані: $migrL^F$ – список файлів для міграції,

Вихідні дані: *MRes* – результат міграції файлів (позитивний/негативний)

1. $i \leftarrow 1$
2. $MRes \leftarrow$ false
3. **while** $\sum_{k=1}^{u_m} s_k - size(migrL^F) < Th^{SSD}$ **then**
4. Мігрувати блоки *i*-го файлу на повільний рівень;
5. Видалити *i*-й файл зі швидкого рівня;
6. Видалити *i*-й файл з L^F та $migrL^F$;
7. $i \leftarrow i + 1$
8. $MRes \leftarrow$ true
9. **end while;**
10. **return** *MRes*

Алгоритм 5.7 має складність O(a), де a – кількість файлів в списку $migrL^F$. Алгоритм міграції даних з повільного рівня на швидкий (Алгоритм 5.8) працює у такий спосіб:

Алгоритм 5.8. Міграція файлу з повільного рівня на швидкий
Вхідні дані: *F* – запитуваний для читання/запису файл,
Вихідні дані: результат міграції файлу (позитивний/негативний)
1. $k \leftarrow 1$, $Mg \leftarrow$ True
2. **repeat**
3. **if** (файл *F* розташований на швидкому рівні) **then**
4. Зчитати/записати блоки файлу зі швидкого рівня
5. **else**
6. Налаштувати зчитування/запис блоків файлу *F* з повільного рівня.
7. Зчитати/записати блоки файлу з повільного рівня
8. **while** ($k \le u_m$) and *Mg* **do**
9. **if** $s_{km} - size(F) > 0$ **then**
10. Записати блоки файлу *F* на швидкісний пристрій *k*
11. Налаштувати зчитування/запис блоків файлу *F* зі швидкісного рівня
12. Видатили блоки файлу *F* з повільного пристрою.
13. $Mg \leftarrow$ False
14. **end if**
15. $k \leftarrow k + 1$
16. **end while**

17. **if** $k = u_m$ and *Mg* **then**
18. Виконати Алгоритм 5.7
19. **end if**
20. **end if**
21. **until** (транзакції з файлом відбуваються)
22. **return** true

Алгоритм 5.8 має складність O(u_m), де u_m – кількість пристроїв швидкого рівня на *m*-му ФС. Алгоритм 5.8працює кожного разу, коли починаються транзакції з певним файлом. При цьому Алгоритм 5.7 може запускатись як у відповідь на нестачу вільного місця на швидкому рівні, так і через певні інтервали часу. Визначення таких інтервалів розглянуто в пп. 3.6, 4.3, 8.5.3.

5.10. Метод реплікації даних між вузлами сховища

Розроблений в дисертації метод управління реплікацією даних у сховищі хмарного ЦОД використовується при всіх стратегіях управління. Вибір вузла зберігання даних для реплікації повинен відбуватись залежно від поточних показників його роботи, таких як кількість вільної оперативної пам’яті, вільного місця на швидкому та повільному рівнях сховища, завантаження мережевого інтерфейсу та ін. Метод реплікації даних показаний на рис. 5.7 і виконується в три етапи:

1) визначення якісних показників кожного серверу;
2) обрання серверів з найкращими показниками за критерієм (16);
3) реплікація даних на обрані сервери.

Перший етап роботи методу виконується сервером метаданих кожного разу, коли нові дані приходять від кожного вузла зберігання даних. Другий і третій етапи виконуються на вузлі, з якого буде створена репліка блоку даних. Для даних, що зберігає сервер метаданих, теж застосовується реплікація, але вже з більш жорсткими вимогами і непарним значенням фактора реплікації (*RF*=5, 7…).

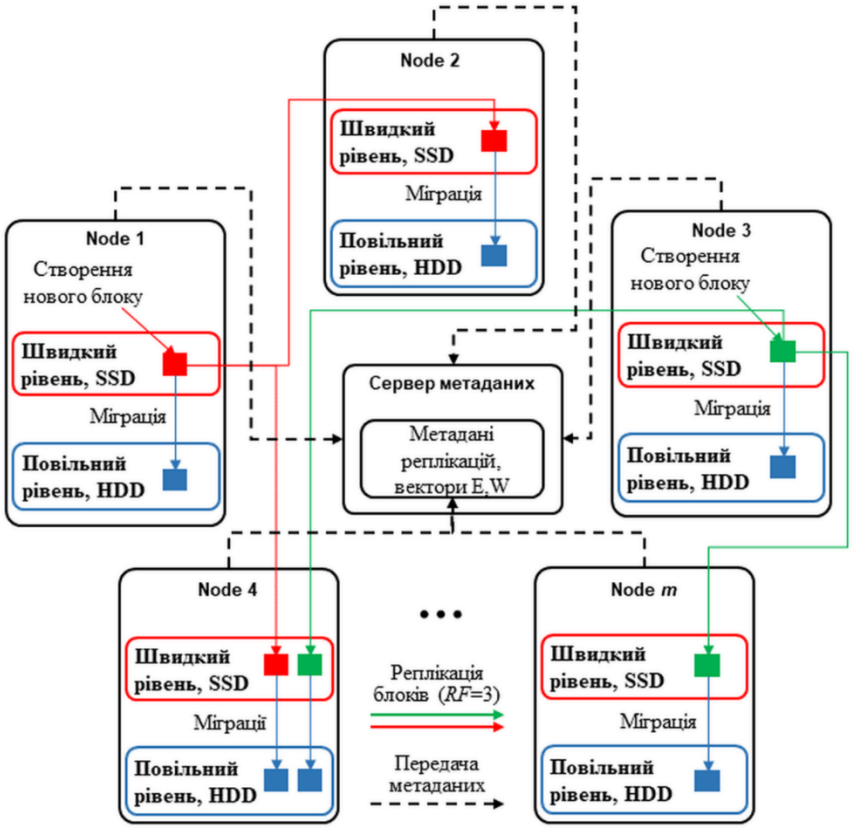


Рис. 5.7. Схема методу реплікації даних

Метод реплікації використовується тільки для блоків даних, які вперше створені на вузлі. Після зміни блоку даних унаслідок транзакції доступу, змінений блок відправляється на вузли зберігання даних, які були визначені раніше. Тобто, змінюються відповідні репліки, які вже розміщені на інших вузлах. При видаленні блоку оновлюється інформація в матриці **R**, видаляються репліки цього блоку, оновлюється інформація векторів **E** і **W**. Репліки блоків даних створюються на повільному рівні поки обсяг вільного місця не стане менше визначеного порогового значення Th^{HDD}. Значення Th^{HDD} залежить від інтенсивності роботи з блоками даних на кожному вузлі і підбирається експериментально.

Визначення вузлів-реплік виконується як ілюструє Алгоритм 5.9. При пошуку вузлів реплікації, для кожного i-го вузла, крім поточного (з якого буде відбуватись реплікація), обчислюється коефіцієнт реплікації p^r за формулою:

$$\begin{aligned} p_i^r &= p^e(1/e_i) + p^w w_i + p^z z_i, \\ p^e + p^w + p^z &= 1 \end{aligned}, \qquad (5.39)$$

де p^w – коефіцієнт, що враховує важливість покажчика кількості транзакцій доступу до блоків даних, p^e – коефіцієнт, що враховує важливість покажчика об'єму вільного місця, p^z – коефіцієнт, що враховує важливість покажчика затримки передачі даних. Алгоритм 5.9 має складність O(M), де M – кількість ФС.

Алгоритм 5.9. Реплікація блоку даних з сервера *m*

Вхідні дані: **W**, **E**, **Z**

Вихідні дані: *rL* – список номерів серверів (вузлів) з реплікою блоку

1. *rL*←NULL
2. *sL*←NULL
3. **for** *i*=1 **to** *M* **do**
4. **if** (e_i<>NULL) and (w_i<>NULL) and (z_i<>NULL) and (i<>m) and (e_i>Th^{HDD}) **then**
5. $sL \leftarrow (i, p^e(1/e_i) + p^w w_i + p^z z_i)$
6. **else** $sL \leftarrow (i, \text{NULL})$
7. *sL.sortIncreasingReplCoef*()
8. **end for**
9. *i*←1, *j*←0
10. **while** *i*<*RF* **do**
11. **if** sL_j<>NULL **then**
12. *rL.add*(sL_j)
13. *i*←*i*+1
14. **end if**
15. *j*←*j*+1
16. **end while**
17. **return** *rL*

Після визначення вузлів-реплік, но кожний вузол зі списку *rL* репліка блоку даних спочатку записується на швидкий рівень, а згодом, якщо блок не змінювався, виконується міграція цього блоку на повільний рівень (Алгоритм 5.7). Також, оновлюється інформація в матриці **R**.

Запропонований метод дає можливість коригувати значення коефіцієнтів у рівнянні (5.39) через оцінку стандартного відхилення на поточному кроці методом [49] і збільшення коефіцієнту для того параметру, значення стандартного відхилення якого зростає. Таким чином, мінімізація критерію (5.38) досягається підбором коефіцієнтів рівняння (5.39).

Реалізація методу управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД у вигляді діаграми класів застосунку моделювання наведена у додатку А.

Висновки до розділу 5

1. Проблема управління хмарним ЦОД з нестачею ресурсів і сталому навантаженні вирішується за допомогою методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу, який реалізує стратегію S_1. Модифікований в дисертації метод імітації відпалу дозволяє отримати близьку до оптимальної карту розподілу ВМ. Таким чином, досягається максимальна продуктивність та повноцінне використання ресурсів кожного ФС ЦОД. Проблема консолідації ВМ розглядається як багатовимірна проблема упаковки в ємності з урахуванням того, що властивості елементів можуть бути змінені динамічно, а також існують інші обмеження при розміщенні ВМ на ФС.

У розділі розроблено два варіанти реалізації стратегії S_1 з використанням алгоритму імітаційного відпалу для вирішення проблеми консолідації нових і вже працюючих ВМ. Для запропонованого модифікованого алгоритму імітаційного відпалу визначено конфігурацію системи, функцію отримання нової конфігурації, цільову функцію для задачі оптимізації. Для зменшення кількості порушень SLA під час міграцій ВМ в алгоритмі OIB враховані обмеження на кількість одночасних міграцій з/на ФС. Крім того, обидва алгоритми OIB дозволяють зарезервувати деякі ресурси ФС для реагування на випадкове збільшення попиту на ресурси у найближчому часі.

2. Для управління ресурсами ЦОД з надлишком ресурсів при сталому навантаженні в рамках стратегії S_2 розроблено двостадійний метод управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку. Проаналізовано роботу евристики першої та другої стадій запропонованого методу, розроблений алгоритм променевого пошуку для вирішення задачі управління ресурсами.

Перерозподіл ВМ виконується з урахуванням обмеження допустимої кількості міграцій. Завдяки урахуванню обмеження кількості міграцій на один ФС запропонований метод може бути застосований в реальних умовах ЦОД.

Унаслідок дослідження пропонується використовувати методику з порогом вільних ресурсів, що показала більш якісні результати консолідації ВМ. Перевагою використання методики з порогом вільних ресурсів є можливість адаптуватися до стану кластера через зміну порогу перед циклом управління, враховуючи інші ресурси, час міграції ВМ та інтенсивність надходження заявок на створення нових ВМ [280].

Основними результатами розділу є: алгоритм пошуку променя для вирішення задачі консолідації ВМ за певних умов; евристики першого та другого етапів алгоритму; функція оцінки; умови виконання алгоритму; розгляд обмежень щодо максимальної кількості одночасних міграцій з/на ФС. Крім того, техніка з порогом вільних ресурсів дозволяє пристосовуватися до стану кластера, враховуючи стан інших ресурсів, час міграції ВМ, інтенсивність вхідних запитів на створення нових ВМ, і обмеження на максимальну кількість одночасних міграцій з/на ФС.

3. Для управління ресурсами ЦОД з нестачею ресурсів і трендом на збільшення навантаження в схемі реалізації стратегій S_5 і S_6 розроблено метод динамічної консолідації і розміщення ВМ на основі алгоритму навчання з підкріпленням. Обґрунтовано можливість застосування методу навчання з підкріпленням для управління ресурсами хмарних ЦОД. Розроблено метод при виборі управляючих впливів враховує витрати електроенергії та кількість порушень SLA. Перевагою методу динамічного розміщення ВМ є здатність виконувати в режимі онлайн розміщення нових ВМ одночасно з перерозподілом вже працюючих ВМ.

Розроблений алгоритм агента, який, через сприймання стану ФС та ВМ, враховує зміну робочого навантаження на ресурси для прийняття рішення щодо включення або переключення в сплячий режим незавантажених ФС з метою зменшення витрат електроенергії. Запропонований агент навчання з підкріпленням базується на підході *Q*-learning, який дозволяє визначати наближену до оптимальної політику управління режимами роботи ФС [281] без створення моделі середовища та попередньої інформації про навантаження.

Запропоновані моделі використовуються для схем реалізації стратегій управління ресурсами хмарного ЦОД.

Запропонований агент НП дозволяє визначити близьку до оптимальної політику управління ФС без створення моделі ЦОД та попередньої інформації про навантаження. У рамках майбутньої роботи планується удосконалити запропонований метод через облік споживання енергії в режимі налаштування/увімкнення ФС. Іншим вдосконаленням є адаптація середовища CloudSim до динамічних змін кількості ВМ під час моделювання.

4. Для управління ресурсами ЦОД з надлишком ресурсів і трендом на збільшення навантаження в схемі реалізації стратегії $\boldsymbol{S_6}$ розроблено метод управління потужністю хмарного ЦОД, який використовує запропоновані метрики, враховує гетерогенність ФС і визначає тип і кількість ФС, які треба увімкнути для обслуговування прогнозованого навантаження. Розроблений метод пропонується використовувати для оцінки нижньої границі кількості ФС, які повинні знаходитися в робочому режимі для розміщення нових ВМ з урахуванням кожного ресурсу, який враховується при розміщенні ВМ і є критичним для її роботи.

5. Розроблено метод управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД на основі кешування за розміром файлу і за кількістю транзакцій доступу до файлу для управління міграцією і на основі кількості транзакцій доступу до блоків даних, об’єму вільного місця та затримки передачі даних між вузлами зберігання даних для управління реплікацією, який використовується в схемах реалізації усіх стратегій управління.

Для управління процесами зчитування, створення, видалення та модифікації даних на рівні сховища ФС запропоновано і досліджено метод управління розподіленим дворівневим сховищем на основі міграції даних між швидким і повільним рівнями сховища та реплікації даних між вузлами зберігання даних.

Запропонований метод управління базується на моделі розподіленого дворівневого сховища з реплікацією і алгоритмах міграції даних між швидким та повільним рівнями сховища за критерієм мінімізації вартості збереження даних.

Вивільнення місця з швидкого рівня відбувається з використанням двох критеріїв: за розміром файлу і за кількістю транзакцій доступу до файлу. З метою забезпечення відмовостійкості системи зберігання даних розроблений метод реплікації даних із заданим фактором реплікації, що враховує кількість транзакцій доступу до блоків даних, об’єм вільного місця та затримку передачі даних між вузлами зберігання даних.

Розроблено модель дворівневого сховища і метод управління дозволяє без суттєвого збільшення вартості зберігання даних підвищити продуктивність операцій читання/запису даних ВМ та контейнерами в гіперконвергентних і хмарних ЦОД. Результати дослідження показують, що використання дворівневих сховищ із запропонованим методом управління дає можливість знизити вартість збереження даних за рахунок зменшення об’єму і кількості пристроїв швидкого рівня [287].

6. Розроблено метод управління міграцією та розміщенням ВМ в сталому режимі з використанням прогнозування в рамках схеми реалізації стратегії S_2, який реалізується адаптивним програмно-визначеним менеджером і включає в себе два основних алгоритми: Алгоритм управління міграцією та розміщенням ВМ і алгоритм роботи екземпляра процесу управління міграцією і розміщенням нових ВМ. Визначено регламент взаємодії АПВМ і МФС за рахунок використання спеціальних тегів і багатомовного обміну повідомленнями.

РОЗДІЛ 6. РОЗВИТОК ДЕКОМПОЗИЦІЙНО-КОМПЕНСАЦІЙНОГО ПІДХОДУ ДЛЯ УПРАВЛІННЯ ІТ-ІНФРАСТРУКТУРОЮ ІНТЕРНЕТУ РЕЧЕЙ НА ОСНОВІ МІКРОХМАРИ

У розділі розроблено узагальнену модель системи інтернету речей і архітектуру системи IoT, що базується на використанні мікрохмари з метою розподілу ресурсів, що забезпечують роботу сервісів в ІТ-інфраструктурі хмарних ЦОД. Розвинуто декомпозиційно-компенсаційний підхід для управління якістю послуг в системі IoT на основі мікрохмари, розподілено функції управління системою IoT між мікрохмарою і застосунками в хмарі, визначені керуючі і координуючі впливи.

6.1. Узагальнена модель системи інтернету речей

IoT системи масштабу міста або великого регіону можуть охоплювати частину міста, області, країни. Враховуючи необхідність розподілу ресурсів між ядром системи та периферійними зонами, архітектуру IoT системи доцільно будувати за схемою, що поєднує хмару (Cloud) та мікрохмари (Micro cloud) [276, 279]. До такого роду систем масштабу міста відносяться, наприклад, системи автоматизації паркування, моніторингу автомобільного трафіку, управління дорожнім рухом, управління камерами спостереження, управління комунальними сервісами та громадським транспортом. У цьому випадку мікрохмара може охоплювати адміністративний район міста або необхідну частину цієї території. Також, залежно від вимог до безпеки, один адміністративний район може бути охоплений декількома мікрохмарами.

Разом з вертикальною взаємодією хмари та мікрохмари пропонується також горизонтальна взаємодія між відповідними мікрохмарами [309] з метою забезпечення:

- мінімальних значень затримки передачі та оброблення даних;
- заданої еластичності ресурсів;
- обміну даними між мікрохмарами в межах доменів безпеки та перерозподілу обчислень;

– мобільності мікрохмари;
– скорочення обсягів трафіку до хмари при великій кількості периферійних вузлів IoT;
– спільного використання даних і управління декількох мікрохмар при використанні розподілених застосувань реального часу.

Загальна модель запропонованої архітектури для системи IoT [309, 310], що базується на використанні *N* мікрохмар та хмари-ядра, показана на рис. 6.1.

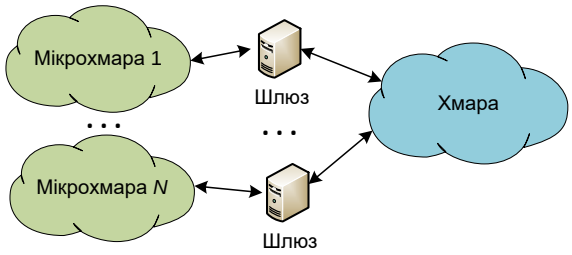


Рис. 6.1. Схема взаємодії N мікрохмар з хмарою-ядром в системі IoT

Схема обміну даними та інформаційної взаємодії між компонентами в системі IoT, що базується на використанні декількох мікрохмар, показана на рис. 6.2 [276].

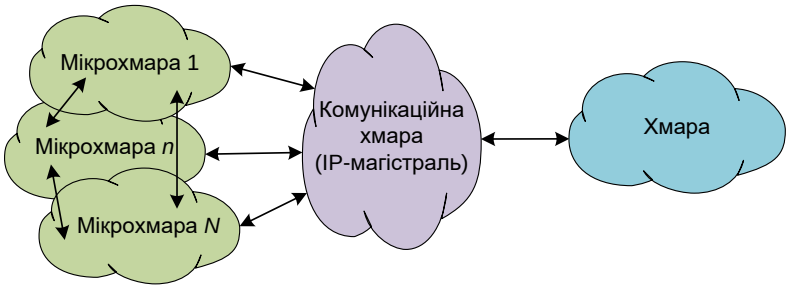


Рис. 6.2. Схема можливих варіантів обміну даними між хмарою і *N* мікрохмарами

Взаємодія давачів та виконавчих пристроїв в складі *n*-ї мікрохмари M_n, $n=\overline{1,N}$, де N – кількість мікрохмар в системі IoT, як між собою, так і з хмарою, може виконуватися через IP-магістраль або безпосередньо між мікрохмарами. В останньому випадку пропонується використовувати мережеві технології Wi-Fi, 3G, 4G, LTE, IEEE 802.15.4, Sensor-Net, WiMAX, Ultra-Wide Band, ZigBee, Bluetooth, 6LoWPAN для безпосередньої взаємодії між мікрохмарами [290].

Структура окремої *n*-ї мікрохмари M_n, $n=\overline{1,N}$ показана на рис. 6.3.

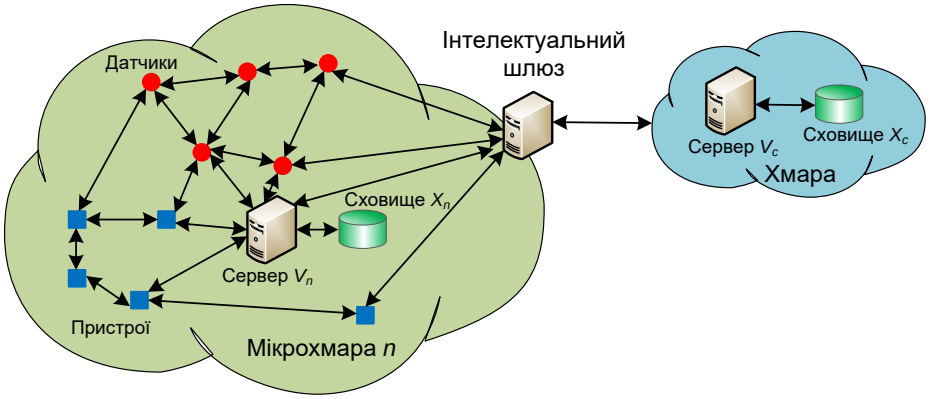


Рис. 6.3. Структура окремої n-ї мікрохмари M_n та взаємодія з хмарою

Мікрохмара M_n включає в себе локальний сервер (фізичний або віртуальний) V_n та локальне сховище X_n.

Взаємодія між елементами мікрохмари здійснюється за принципами самоорганізації в тимчасовій мережі, що спроектована і функціонує з урахуванням вимог технології зеленої мережі (англ. Green networking technology). Причому всі пристрої мікрохмари повинні задовольняти вимогам високої доступності із заданою затримкою виконання функції при включенні пристрою. У кожній мікрохмарі може не бути жодного або міститися один або декілька локальних серверів та сховищ залежно від вимог до зберігання та оброблення даних. При відсутності в мікрохмарі локального сервера обчислювальні задачі і зберігання даних цієї мікрохмари виконується серверами і сховищами сусідніх мікрохмар або сервером і сховищем, що знаходяться в хмарі, як це показано на рис. 6.4 [276].

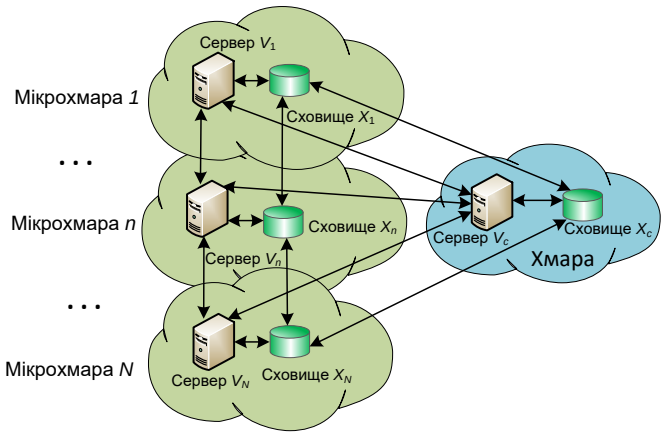


Рис. 6.4. Схема взаємодії локальних серверів і систем зберігання даних мікрохмар і хмари

Локальні сервери і сховища мікрохмари у взаємодії з одним або декількома серверами в хмарі забезпечують функціонування розподілених додатків. Робота розподілених додатків спрямована на надання сервісу S_m, $m=\overline{1,M}$, де M – кількість сервісів в системі IoT, що функціонує на базі хмари та мікрохмар.

При цьому необхідно вирішувати такі задачі:

- оптимально розподілити завдання між мікрохмарами M_n, $n=\overline{1,N}$ і хмарою;
- розподілити функції між мікрохмарами і хмарою;
- розподілити ресурси між мікрохмарами і хмарою;
- оптимізувати інфраструктуру взаємодії мікрохмар та хмари, виходячи з вимог до сервісів та вартості ресурсів;
- виконати умови масштабування ресурсів та забезпечення еластичності;
- врахувати мобільність сенсорів та пристроїв;
- визначити та вести розрахунки значень метрик для обміну даними між мікрохмарами та хмарою;
- визначити та застосувати методи аналізу даних, аналітики реального часу, в тому числі для Big data;
- виконати декомпозицію та децентралізацію процесів управління;
- розробити принципи побудови та структуру системи управління хмарною інфраструктурою IoT;
- визначити методи управління безпекою, включаючи управління доступом, стійкістю до атак, шифрування, ідентифікації та захисту персональних даних;
- розробити інформаційну технологію і інструментарій для опису сервісів, об'єктів моніторингу та управління і інших компонентів IoT інфраструктури на основі мікрохмар та хмари [279].

6.2. Архітектура системи інтернету речей, що базується на концепції мікрохмари

Системи IoT масштабу міста пропонується умовно розділити на два великі класи: системи моніторингу і системи моніторингу та управління. Системи моніторингу включають в себе міські системи, такі як системи моніторингу дорожнього руху, камери відеоспостереження, системи відстеження надзвичайних ситуацій, системи екологічного моніторингу і мікроклімату, системи моніторингу на робочих місцях та у приватних будинках. Основною задачею цих систем є збір даних для систем управління та прийняття рішень міського призначення. До систем моніторингу та управління відносяться міські системи, такі як системи паркування, управління громадським транспортом, надання послуг охорони здоров'я, управління ланцюжками поставок, системи пожежогасіння [279].

Структура системи IoT на основі мікрохмари, що здійснює глобальний моніторинг в масштабах міста, і напрямки потоків передачі інформації в такій системі розроблені у працях [276, 309] і наведені на рис. 6.5.

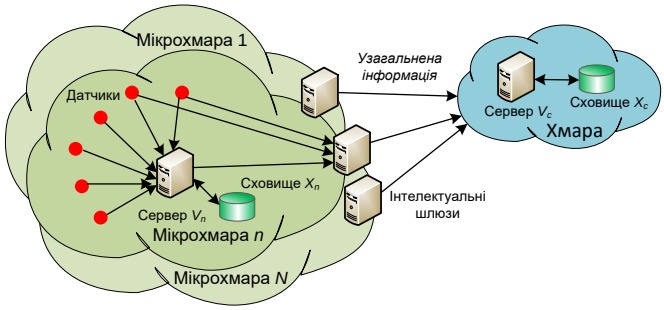


Рис. 6.5. Структура глобального моніторингу в системі IoT на основі мікрохмари та хмари

На Рис. 6.5 в кожній n-й мікрохмарі дані надходять з давачів, обробляються в сервері V_n і запам'ятовуються в сховище X_n. Узагальнена інформація результатів моніторингу в окремій мікрохмарі, отримана після оброблення в сервері V_n, надходить на сервер хмари V_c і в сховище хмари X_c. Сервери хмари, обробляючи інформацію моніторингу від усіх мікрохмар, надають можливість отримати глобальну картину в масштабах міста або регіону про стан

контрольованих параметрів. Частина давачів і пристроїв можуть безпосередньо передавати дані в сервер хмари [276].

Одна з найважливіших задач, яку потрібно вирішити при створенні системи IoT на основі мікрохмари та хмари, полягає у визначенні ступеня узагальненості інформації, що передається з мікрохмар до хмари. Передача всієї інформації, що надходить з давачів без попереднього оброблення локальним сервером в мікрохмарі може призвести до надмірної завантаженості каналів зв'язку. Велика ступінь узагальнення після оброблення в локальному сервері мікрохмари призводить до скорочення обсягів даних, що передаються в хмару, але скорочення надмірності даних може привести до зниження точності глобального моніторингу, а попередня обробка сервером мікрохмари збільшує затримку отримання даних в хмарі [276].

Винятком є випадок, коли дані моніторингу не виходять за межі мікрохмари, обробка даних моніторингу повністю здійснюється в мікрохмарі, а в хмару передаються тільки дані, що перевищують встановлені пороги, а також повідомлення про надзвичайні або аварійні ситуації [279].

Структура системи IoT на основі мікрохмари, що здійснює глобальний моніторинг і локальне управління в зоні покриття мікрохмари, наведена на рис. 6.6 [276].

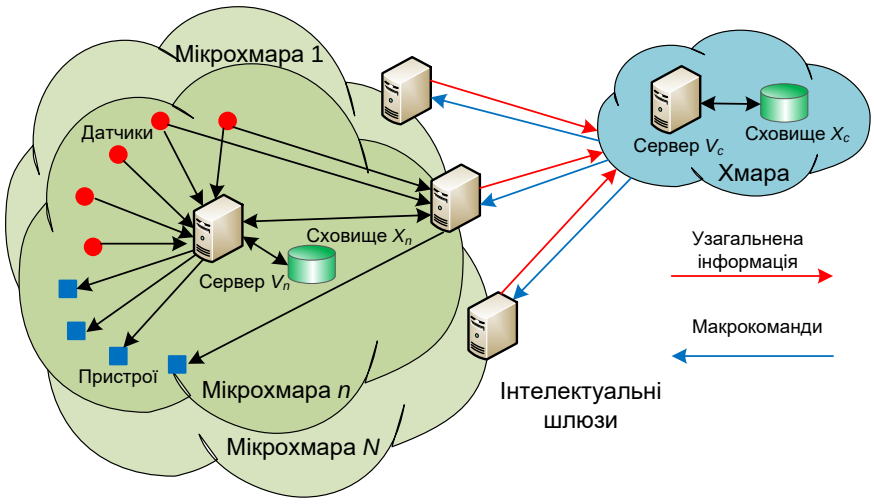


Рис. 6.6. Взаємодія локальних серверів і систем зберігання даних в мікрохмарі та хмарі

В системі глобального моніторингу і локального управління, що зображена на рис. 6.6, в кожній мікрохмарі здійснюється збір і попередня обробка даних

моніторингу, що надходять з давачів, розташованих в зоні дії мікрохмари. Узагальнена інформація з мікрохмари надходить на сервер хмари, який формує макрокоманди або політики управління на основі даних, що отримані від усіх мікрохмар. Сформовані макрокоманди або політики управління надходять до мікрохмар, де сервери мікрохмар формують команди управління для локальних пристроїв. На частину локальних пристроїв команди управління можуть надходити безпосередньо з сервера хмари V_c. [276].

6.3. Удосконалення декомпозиційно-компенсаційного підходу щодо управління якістю послуг в системі інтернету речей на основі мікрохмари

Вирішення задач в системах IoT, що побудовані за архітектурою взаємодії мікрохмара-хмара, вимагає обчислювальних ресурсів і ресурсів зберігання даних. Такі ресурси є і в мікрохмарі, і в хмарі. Ресурси мікрохмари зазвичай обмежені, тому виникає проблема розроблення методів розподілу завдань між мікрохмарою і хмарою для ефективного використання наявних в них ресурсів [276].

Для вирішення таких завдань при управлінні корпоративними IT-інфраструктурами в [69] запропонований декомпозиційно-компенсаційний підхід. У дисертаційній роботі декомпозиційно-компенсаційний підхід розвинутий [274, 276, 279] з метою вирішення завдань розподілу і перерозподілу ресурсів в системі IoT з архітектурою мікрохмара-хмара (МХ-Х) в такий спосіб.

Для забезпечення рентабельності бізнесу компаній, що працюють в сфері IoT, необхідно, щоб вони отримували множину $\mathcal{G} = \{s_i\}$, $i = \overline{1,K}$ необхідних послуг IoT з максимальною якістю $\mathcal{Q}$ і мінімальними витратами $\mathcal{C}$.

Управління рівнем послуг в системі IoT з архітектурою мікрохмара-хмара пропонується здійснювати інтегрованою взаємодією трьох процесів: узгодження рівня послуг, планування ресурсів і управління рівнем послуг, як це показано на рис. 6.7 [279].

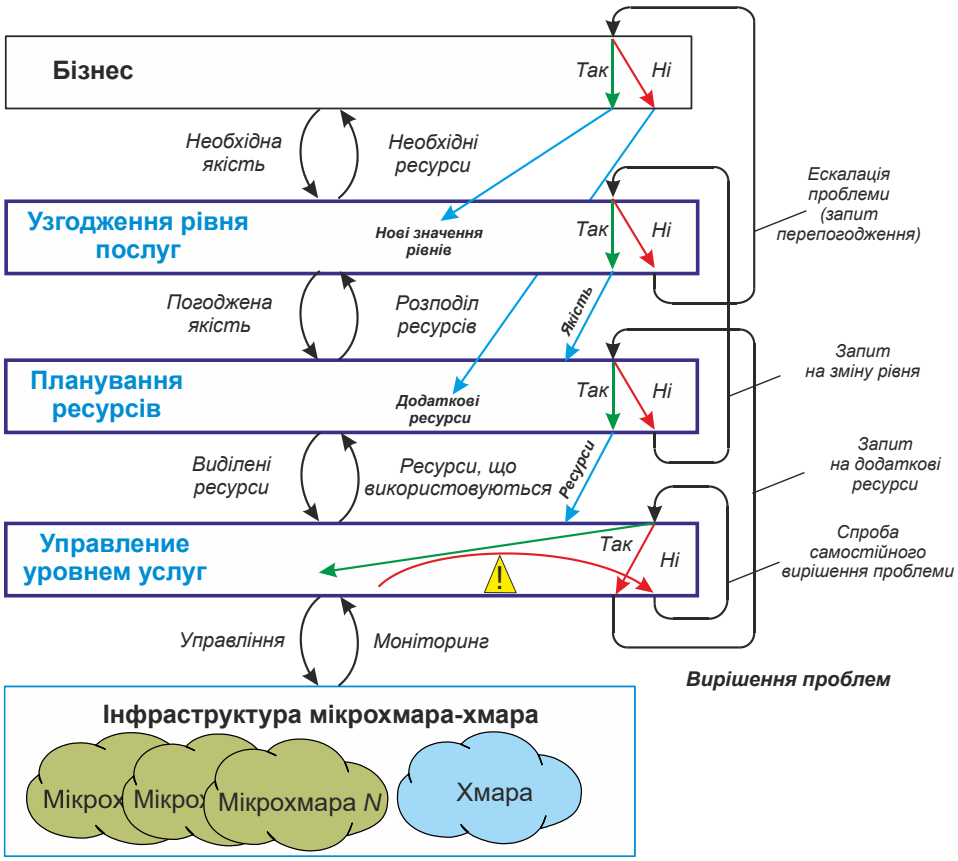


Рис. 6.7. Взаємодія процесів при управлінні рівнем послуг в системі IoT з архітектурою мікрохмара-хмара

Процес узгодження рівня послуг запускається з ініціативи бізнес-менеджерів системи IoT або замовників сервісу IoT і закінчується формуванням або оновленням елементів множини $\mathcal{G}$ і матриці $Q = \|q_{ki}\|$, елемент q_{ki}, $k = \overline{1, L_i}$, $i = \overline{1, K}$, якої відповідає значенню k-го показника якості i-ї послуги. Для функціонування послуг $\mathcal{G}$ їм виділяється сумарна кількість ресурсів r в системі IoT як у мікрохмарі, так і в самій хмарі. Отримане унаслідок ресурсне забезпечення IoT у вигляді системи $< Q, r >$ є основою для вирішення завдань на рівні нижче [276].

Процес планування полягає у виділенні і закріпленні за кожною послугою s_i, $i = \overline{1, K}$ частини ресурсів мікрохмари та хмари з ресурсів $R_1,...,R_m$, виділених для підтримки послуг. При цьому обсяги або кількість $r_1,...,r_m$, ресурсів $R_1,...,R_m$ відповідно, визначаються у такий спосіб: $r_j = \sum_1^N r_j^{(M_n)} + r_j^{M_c}$, де $r_1^{(M_n)},...,r_m^{(M_n)}$ – кількість ресурсу R_j, $j = \overline{1, m}$, що виділені в мікрохмарі M_n, $r_j^{M_c}$ – кількість

ресурсу R_j, $j=\overline{1,m}$, виділених у хмарі. Одиниця j-го ресурсу має вартість c_j. При цьому $c_j^{M_n}$ – вартість одиниці j-го ресурсу в мікрохмарі M_n, $c_j^{M_c}$ – вартість одиниці j-го ресурсу в хмарі. Тоді кількість всіх ресурсів обчислюється як $r=\sum_{j=1}^{m} r_j$, а вартість c ресурсів визначається у такий спосіб: $c=\sum_{j=1}^{m} r_j \cdot c_j$ [276].

Використання певними послугами призначених ресурсів задається матрицею $\mathrm{P}=\left\|\rho_{ij}\right\|$, де ρ_{ij} дорівнює кількості виділеного послузі s_i ресурсу R_j, $j=\overline{1,m}$, або 0, якщо ресурс не потрібен.

Процес управління рівнем послуг здійснює управління системою IoT так, щоб фактичні значення q_{ki}^{*}, $k=\overline{1,L_i}$, $i=\overline{1,K}$ показників якості послуг відповідали узгодженим значенням з матриці Q, тобто, щоб виконувалася рівність

$$q_{ki}-q_{ki}^{*}=0,\quad k=\overline{1,M_i}, i=\overline{1,K}. \tag{6.1}$$

При невиконанні умови (6.1) визначаються елементи матриці фактичних значень показників якості $Q^{*}=\left\|q_{ki}^{*}\right\|$, для яких $q_{ki}^{*}<q_{ki}$, $k=\overline{1,L_i}, i=\overline{1,K}$. Система управління намагається вирішити задачу на нижньому рівні, змінюючи значення параметрів функціонування системи IoT або перерозподіляючи ресурси між мікрохмарою та хмарою так, щоб збільшити значення q_{ki}^{*} [276].

Якщо унаслідок відновлювальних заходів вдалося забезпечити виконання рівності (6.1), то функціонування системи IoT триває з новими налаштуваннями. Якщо повноважень нижнього рівня недостатньо для досягнення (6.1), то здійснюється ескалація проблеми на рівень планування ресурсів [276].

Процес планування ресурсів намагається вирішити проблему, використовуючи такі механізми:

– виділення в мікрохмарі або хмарі додаткових ресурсів $R_1,...,R_m$ для послуги s_i, для якої виконується умова $q_{ki}^{*}<q_{ki}$. Для цього здійснюється локалізація конкретної мікрохмари або хмари, ресурси яких виявляються перевантаженими при наданні послуги s_i. Якщо в певній мікрохмарі або хмарі може бути виділений додатковий обсяг ресурсу, нестача якого

послужила причиною зниження якості послуги s_i, то формується матриця з новими значеннями елементів $\mathrm{P}' = \left\| \rho'_{ij} \right\|$, причому $\rho'_{ij} > \rho_{ij}$, $j = \overline{1,m}$, або 0, якщо j-й ресурс не потрібен. Таким чином, для підтримки послуги s_i в мікрохмарі або хмарі виділяється додатковий обсяг ресурсу;

- якщо додаткові ресурси в даній мікрохмарі відсутні, то рівень планування ресурсів намагається виділити додаткові ресурси в інших мікрохмарах і переспрямувати в них частину завдань управління IoT, які до цього вирішувалися в проблемній мікрохмарі, передаючи туди ж для оброблення і дані моніторингу, які збираються в проблемній мікрохмарі;
- якщо неможливо задіяти ресурси, які є доступними в інших мікрохмарах, то рівень планування ресурсів робить спробу виділити додаткові ресурси в хмарі;
- іноді в хмарі неможливо виділити додаткові ресурси. Це відбувається або через високу вартість додаткових ресурсів, що є малоймовірним, або через те, що в якості хмари використовується корпоративний обчислювальний центр власника системи IoT, ресурси якого зазвичай обмежені. Якщо в хмарі немає можливості виділити додаткові ресурси, то рівень планування ресурсів намагається зробити перерозподіл ресурсів між послугами, віддаючи ресурси більш важливим послугам за рахунок менш важливих. Якщо проблему вдається вирішити, то значення матриці $\mathrm{P}' = \left\| \rho'_{ij} \right\|$ з новим планом закріплення ресурсів надходять на рівень управління рівнем послуг;
- якщо рішення проблеми на рівні планування ресурсів неможливе, проводиться ескалація проблеми на вищерозташований рівень узгодження рівня послуг [276].

Необхідно відзначити, що коли ресурсів в мікрохмарі не вистачає, і задіюються ресурси або інших мікрохмар або хмари, то при цьому необхідно перерозподілити функції і завдання в інфраструктурі мікрохмара-хмара. Однак це може призвести до погіршення якості роботи мікрохмари і якості надання

послуг системою IoT. При цьому можливий перерозподіл принципів вирішення завдань в мікрохмарі, коли частина високопріоритетних завдань виконується з високою якістю, а інші завдання виконуються з низькою якістю або зберігають свою функціональність з мінімально допустимою якістю [276].

Процес узгодження рівня послуг з ініціативи процесу планування здійснює перегляд спочатку значення q_{ki}, для якого $q_{ki}^{*} < q_{ki}$, а потім, можливо, і значень всіх елементів q_{ki}, $k = \overline{1, L_i}$, $i = \overline{1, K}$ матриці якості послуг Q у бік зменшення. Якщо вдається сформувати матрицю $Q' = \|q'_{ki}\|$ з новими значеннями показників якості послуг, то вона передається на рівень нижче, де проводиться вивільнення ресурсів і виділення їх для послуг, для яких виконується умова $q_{ki}^{*} < q_{ki}$, $k = \overline{1, L_i}$, $i = \overline{1, K}$ [276].

Якщо процес узгодження рівня послуг не має повноважень для формування матриці $Q' = \|q'_{ki}\|$, то проводиться ескалація проблеми на рівень бізнесу, який змушений або згенерувати матрицю $Q' = \|q'_{ki}\|$, з новими значеннями, або збільшити загальний обсяг ресурсів, що призводить до збільшення значень $r_1,...,r_m$ [276].

Процеси, що виконуються при узгодженні рівня послуг, планування ресурсів і управління рівнем послуг, детально описані в роботі [69].

6.4. Розподіл функцій управління системою інтернету речей

У зв'язку з тим, що на сьогодні дослідження теорії управління загострені на багатооб’єктності і великій розмірності, виникає потреба виділення спеціального класу багатооб’єктних розподілених систем управління, до яких відносяться системи управління для IoT, побудованих за архітектурою MC-C. Одним з найважливіших питань, що вирішуються при проектуванні і експлуатації таких систем управління, які є ієрархічними, крім розроблення архітектури, є завдання прийняття рішень. Основні результати дослідження і розроблення функцій управління системою IoT представлені у працях [276, 310].

Складність прийняття рішень в ієрархічних системах управління обумовлена тим, що рішення приймаються на більшості рівнів ієрархії системи управління, а також – обмеженням часу на прийняття рішення. У [249] запропоновано такі системи розглядати як дворівневі системи управління з координатором [70].

Підставою для виділення в системі управління двох рівнів є те, що при управлінні рівнем послуг в архітектурі MC-C система управління функціонує в різних режимах в умовах невизначеності, неповноти і недостовірності інформації, наявності факторів ризику, множини конфліктуючих критеріїв і цілей підсистем системи управління.

За допомогою таких систем управління дуже складно досягти оптимальне функціонування систем IoT, побудованих за архітектурою MC-C. Від таких систем управління потрібне поліпшення якісних характеристик роботи системи IoT. У таких випадках виправданою є побудова дворівневих систем з координатором [249, 70], коли координатор узгоджує самостійні рішення і дії підсистем системи управління для поліпшення роботи системи IoT по архітектурі MC-C в цілому з точки зору якості послуг, що надаються. При цьому дії координатора повинні бути спрямовані на поліпшення глобальної функції якості надання послуг, а прийняття ним рішень здійснюється в умовах невизначеності.

На рис. 6.8 приведена структура системи управління, що здійснює управління рівнем послуг в IoT по архітектурі MC-C, у вигляді дворівневої системи з координатором [249, 70].

Розташування управляючих підсистем (УП) відображає ієрархічну структуру системи управління. Модель складається з вище розташованої по ієрархії керуючої підсистеми ($УП_0$) - координатора та $N + 1$ нижчих керуючих підсистем ($УП_1$,..., $УП_N$, $УП_C$), безпосередньо керують процесом P, який протікає в MC-C інфраструктурі.

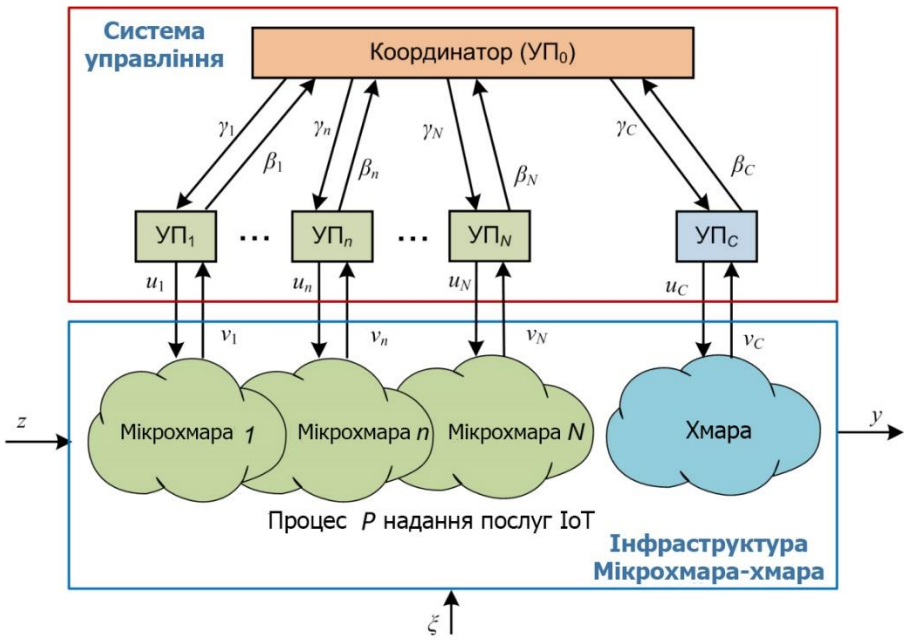


Рис. 6.8. Подання системи управління процесом надання послуг в IoT по архітектурі Micro cloud and Cloud у вигляді дворівневої системи з координатором

Взаємодія УП по вертикалі здійснюється у такий спосіб.

Команди, сигнали або вплив (входи) $\gamma_1,...,\gamma_N,\gamma_C$, що передаються від $УП_0$ до $УП_1,...$, $УП_N$, $УП_C$ є координуючими. Командні сигнали або впливи (входи) $(u_1,...,u_N,u_C)$ від $УП_1,...$, $УП_N$, $УП_C$ до процесу P є керуючими. Знизу вгору надходять сигнали зворотного зв'язку або інформаційні сигнали $v_1,...,v_N,v_C$ і $\beta_1,...,\beta_N,\beta_C$.. Опис дворівневої СУІ може бути здійснений за допомогою термінальних змінних (входів і виходів). У цьому випадку УП описуються як функціональні підсистеми, виходи яких однозначно визначаються входами [70].

Процес P можна описувати як керовану підсистему, на яку впливають керуючі сигнали u від $УП_1,…,УП_N,УП_C$, $u \in U$, U - множина керуючих впливів; надходять вхідні сигнали z, $z \in Z$, що представляють собою запити користувачів; і сигнали ξ, $\xi \in \Xi$, що є збурюючими впливами.

До збурюючих впливів Ξ відносяться несправності в МС-С інфраструктурі, функціональні відмови ресурсів МС або С, запити до інших систем IoT, що спільно використовують ресурси МС-С з розглянутою системою IoT, збільшення обсягів даних моніторингу, що вимагають оброблення і інші причини. Запити до інших систем IoT, розглядаються як впливи по відношенню до даної системи IoT, що ускладнює досягнення цілей управління.

Виходом процесу P є y, $y \in Y$, де Y - множина виходів процесу P, якими є відповіді даної системи IoT на запити користувачів або результати діяльності системи IoT.

Процес P можна представити у вигляді відображення на основі декартова добутку:

$$P : U \times Z \times \Xi \to Y. \tag{6.2}$$

Множина U керуючих сигналів, що впливають на процес P з боку УП$_1$,…,УП$_N$,УП$_C$, зручно представляти у вигляді декартового добутку N+1 множин [70]

$$U = U_1 \times U_2 \times ... \times U_N \times U_C. \tag{6.3}$$

При цьому кожна керуюча підсистема з УП$_1$,…,УП$_N$,УП$_C$ володіє повноваженнями вибору окремої компоненти u_1,…,u_N, u_C керуючого впливу u з відповідної множини U_1,…U_N або U_C для здійснення безпосереднього впливу на процес P.

На входи кожної g-ї керуючої підсистеми УП$_1$,…,УП$_N$,УП$_C$ надходять два сигнали: координуючий сигнал $\{\gamma_1,...,\gamma_N,\gamma_C\} \in \Gamma$ від УП$_0$ і інформаційний сигнал зворотного зв'язку $\nu_1,...,\nu_N,\nu_C$ у вигляді даних моніторингу. Керуючим виходом УП$_1$,…,УП$_N$,УП$_C$ є вплив, обираний УП$_1$,…,УП$_N$,УП$_C$ з відповідної множини U_1,…U_N або U_C. Припустимо, що кожна з УП$_1$,…,УП$_N$,УП$_C$, реалізує відповідне відображення C_1,…,C_N, C_C, таке, що

$$C_1 : \Gamma \times V_1 \to U_1,...,C_N : \Gamma \times V_N \to U_N, C_C : \Gamma \times V_C \to U_C, \tag{6.4}$$

де кожна компонента з V_1,…,V_N,V_C – це множина даних моніторингу v_1,…,v_N, або v_C, що надходять в систему управління з МС-С інфраструктури, причому $v_1 \in V_1,...,v_N \in V_N, v_C \in V_C$. Дані моніторингу v_1,…,v_N,v_C є сигналами зворотного зв'язку для локального контуру управління, основою якого є УП$_1$,…,УП$_N$,УП$_C$.

Сигнали зворотного зв'язку v_1,…,v_N,v_C, що надходять на вхід УП$_1$,…,УП$_N$,УП$_C$, отримані унаслідок моніторингу МС-С інфраструктури. Вони містять інформацію щодо перебігу процесу P. Природно, ці сигнали функціонально залежать від керуючих сигналів u, входів z, збурень ξ і виходів y. Цю залежність можна представити сукупністю відображень [70]

$$f_1 : U \times Z \times \Xi \times Y \to V_1, ..., f_N : U \times Z \times \Xi \times Y \to V_N, f_C : U \times Z \times \Xi \times Y \to V_C. \quad (6.5)$$

Управляюча підсистема $УП_0$ є координатором і виробляє координуючі сигнали $\gamma_1, ..., \gamma_N, \gamma_C \in \Gamma$, причому кожен сигнал з відповідного виходу $УП_0$ надходить тільки на вхід окремої нижчерозташованої керуючої підсистеми $УП_1, \ldots, УП_N$ або $УП_C$. Координатор $УП_0$ виробляє сигнал на основі аналізу інформації, що надходить на його вхід від $УП_1, \ldots, УП_N$ і $УП_C$, і представляє собою сигнали зворотного зв'язку і узагальнену інформацію про стан і функціонування МС-С інфраструктури. У такому разі можна вважати, що координатор реалізує відображення C_0 таке, що

$$C_0 : B \to \Gamma, \quad (6.6)$$

де B - множина інформаційних сигналів β, що реалізують зворотний зв'язок. Причому $\beta = (\beta_1, ..., \beta_N, \beta_C)$ являє собою сукупність сигналів $\beta_1, ..., \beta_N, \beta_C$ зворотного зв'язку, що надходять в координатор $УП_0$ від підсистем $УП_1, \ldots, УП_N$ і $УП_C$.

Аналогічно (6.5) сигнал зворотного зв'язку β, що надходить в $УП_0$, несе в собі інформацію про стан всіх нижчих підсистем, тому він визначається відображенням

$$f_0 : \Gamma \times V \times U \to B, \quad (6.7)$$

де $V = V_1 \times ... \times V_n$. Таким чином, B є функцією координуючих сигналів $\gamma_1, ..., \gamma_N, \gamma_C$, сигналів зворотного зв'язку $v = (v_1, \ldots, v_N, v_C)$, що поступають в $УП_1, \ldots, УП_N, УП_C$, і керуючих впливів $u = (u_1, ..., u_N, u_C)$.

На моделі, зображеної на рис. 6.8, не показано в явному вигляді взаємодія між підсистемами $УП_1, \ldots, УП_N, УП_C$, так само, як і не показано безпосередній вплив $УП_0$ на функціонування МС-С інфраструктури та отримання координатором $УП_0$ сигналів зворотного зв'язку безпосередньо від елементів МС-С інфраструктури, що має місце в реальних системах управління.

Відповідно до [70] координація полягає у впливі на підсистеми управління $УП_1, \ldots, УП_N, УП_C$, що змушує діяти їх узгоджено, підпорядковуючи дії кожної з них єдиній політиці, орієнтованої на досягнення глобальної мети системи,

незважаючи на те, що ця мета може суперечити локальним цілям підсистем. Координацію здійснює $УП_0$, причому саме координатор повинен подолати протиріччя між локальними цілями підсистем $УП_1,…,УП_N,УП_C$.

Успішність діяльності координатора по організації узгоджених дій $УП_1,…,УП_N,УП_C$ оцінюється тим, наскільки успішно досягається глобальна мета управління МС-С інфраструктурою. Досягнення мети координатором можна розглядати як рішення задачі, яка формалізується як завдання прийняття рішення і полягає в оцінці результативності координації. Оскільки ця задача визначається щодо всіх підсистем, включаючи процес *P*, то вона називається глобальною розв'язуваною задачею [70].

Для дворівневих систем повинні бути забезпечені координованість по відношенню до задачі, розв'язуваною $УП_0$, і координованість по відношенню до глобальної задачі [70]. Перше означає, що сигнали $УП_0$ координують вплив на завдання, які вирішуються $УП_1,…,УП_N,УП_C$, а друге, що координатор здатний впливати на $УП_1,…,УП_N,УП_C$ так, що їх спільний вплив на процес *P*, що виконується системою IoT, направлений на вирішення глобальної задачі.

Успішне функціонування системи управління, що відповідає дворівневої моделі, може бути забезпечено тільки тоді, коли цілі підсистем узгоджені між собою і узгоджені з глобальною метою системи [249, 70].

У дворівневій системі виділяють три типи цілей (завдань): глобальна мета, мета координатора $УП_0$, мета підсистем $УП_1,…,УП_N,УП_C$. В основному на процес *P* безпосередньо впливають тільки $УП_1,…,УП_N,УП_C$, тому глобальна мета може бути досягнута тільки опосередковано через дії $УП_1,…,УП_N$, $УП_C$, які повинні бути скоординовані щодо глобальної мети, а також цілі координатора.

Глобальна мета – підвищення ефективності виконання бізнес-процесів, виходить за рамки безпосередньої діяльності дворівневої системи, наведеної на рис. 6.8, і жодна з підсистем $УП_1,…,УП_N,УП_C$ не орієнтована на досягнення глобальної мети або рішення глобального завдання. Глобальне завдання може бути вирішене тільки спільними діями всіх керуючих підсистем $УП_1,…,УП_N,УП_C$.

Глобальна мета. З огляду на той факт, що МС-С інфраструктури створюються для підвищення ефективності функціонування систем IoT, глобальну мету системи управління можна визначити як забезпечення максимальної якості сервісів системи IoT з мінімальними витратами $\mathcal{C}$. Так, метою процесного управління відповідно до ITSM і ISO є постійне підвищення рівня ІТ-послуг, що формально можна записати як $\max \mathcal{Q}$.

Максимальна якість надання послуг в МС-С інфраструктурі буде досягатися в тому випадку, коли

$$\max \mathcal{Q} \Leftrightarrow \max Q_i, \forall i = \overline{1, K} \Leftrightarrow \max q_{ki}, \forall i = \overline{1, K}, \forall k = \overline{1, L_i}, \qquad (6.8)$$

де $Q_i, i = \overline{1, K}$ – якість *i*-ї послуги; $q_{ki}, k = \overline{1, L_i}$ – значення *k*-го показника якості *i*-ї послуги, що надається системою IoT.

Для досягнення мети процесного управління необхідно безперервно нарощувати ресурси МС-С інфраструктури, що є неприйнятним, перш за все, з економічної точки зору. З іншого боку, підвищення економічної ефективності ведення бізнесу вимагає скорочення витрат на МС-С інфраструктуру, тобто дій, націлених на досягнення $\min \mathcal{C}$.. Підтримка якості послуг на цьому рівні є основним завданням координатора.

Мета координатора. Метою координатора є підтримка якості послуг $\mathcal{Q}$ на узгодженому рівні з мінімальними витратами $\mathcal{C}$ на використані ресурси. Мету координатора можна формалізувати у такий спосіб

$$\mathcal{Q} = \text{const}\big|_{\min \mathcal{C}}. \qquad (6.9)$$

Вираз (6.6) означає, що координатор з усіх можливих втручань буде вибирати такі, які вимагають мінімальної вартості реалізації.

Вимога підтримки узгодженого рівня послуг стосується всіх послуг і окремих показників якості послуг:

$$\mathcal{Q} = \text{const} \Leftrightarrow Q_i = \text{const}, \forall i = \overline{1, K} \Leftrightarrow q_{ki} = \text{const}, \forall k = \overline{1, L_i}, \forall i = \overline{1, K}. \qquad (6.10)$$

Тут необхідно обумовити таку обставину. Основним способом підвищення якості *i*-ї послуги, $i = \overline{1, K}$, є виділення додаткових ресурсів застосунків, що підтримують роботу *i*-ї послуги. При перевищенні рівнем *i*-ї послуги цільового

значення проводиться скорочення ресурсів, виділених відповідним застосункам, як цього вимагає критерій $\min \mathcal{C}$. У той же час, останній сервер, що надає *i*-у послугу, не може бути відключений, незважаючи на те, що якість цієї послуги як і раніше вище необхідного, оскільки це призведе до повного припинення надання послуги. Таким чином, завжди буде якийсь фіксований мінімум витрат $\mathcal{C}$, після чого подальше скорочення витрат буде неможливо.

Локальні цілі. Метою локального управління є підтримання заданих значень параметрів функціонування МС-С інфраструктури з мінімальними витратами. У моделі системи управління, наведеній на рис. 6.8, керуючі підсистеми $УП_1,…,УП_N,УП_C$ можуть мати власні цілі функціонування.

6.5. Принцип координації і синтез координатора в системі інтернету речей

Виконання вимог координованості і сумісності виступають обмеженнями при визначенні стратегій, якими може керуватися координатор. Запропоновані в [70] принципи координації, засновані на постулаті сумісності, не можуть бути використані в системі управління МС-С інфраструктурою, оскільки вони припускають або отримання і використання точного прогнозу значень параметрів процесу *P*, або вимагають знання виду функцій або аналітичних виразів для вирішення завдання координації. Для вирішення проблеми координації необхідно після декомпозиції глобальної задачі провести синтез координатора і визначитися з методами, процедурами або алгоритмами координації. Основні результати з розроблення координатора наведені у працях [279, 310].

Сформулюємо мету координатора (6.10) в такому вигляді:

$$\begin{aligned}&\min \Delta Q_i = \min(Q_i - Q_i^*),\ \forall i = \overline{1, K} \Leftrightarrow \min \Delta q_{ki} = \min(q_{ki} - q_{ki}^*),\\ &\forall k = \overline{1, L_i}, \forall i = \overline{1, K},\end{aligned} \tag{6.11}$$

де Q_i і Q_i^* - цільове і фактичне значення якості *i*-ї послуги; q_{ki} и q_{ki}^* - цільове і фактичне значення показника якості *i*-ї послуги, причому фактична якість вважається гірше необхідної при $Q_i > Q_i^*$ і, відповідно, при $q_{ki} > q_{ki}^*$.

На входи процесу P надходять керуючі і збурюючі впливи, а завдання системи управління зводиться до вибору управління, протидіє обуренню. Виходячи з (6.11), координатор повинен порівнювати поточні значення показників якості послуг $q_{ki}^{*}, k=\overline{1,L_i}, i=\overline{1,K}$, що надаються користувачам процесом P, з цільовими значеннями $q_{ki}, k=\overline{1,L_i}, i=\overline{1,K}$ і виробляти координуючі сигнали, які мінімізують відхилення.

При управлінні, спрямованому на підтримку узгодженого рівня якості надання послуг, доцільним є використання принципу управління по відхиленню [249]. Виходи процесу P проходять відповідні перетворення, які полягають у зведенні метрик для визначення значень $q_{ki}, k=\overline{1,L_i}, i=\overline{1,K}$. У цьому разі, виходи процесу P по ланцюгу зворотного зв'язку надходять в координатор, де порівнюються з цільовими значеннями. На підставі відхилення $\Delta q_{ki}, k=\overline{1,L_i}, i=\overline{1,K},$ виробляються координуючі сигнали для УП$_1$,…,УП$_N$,УП$_C$.

В системі управління можуть бути виміряні основні збурюючі впливи, до яких відноситься кількість $\hat{a}=\{a_l, l=\overline{1,I}\}$ користувачів послуг, вплив несправностей на якість сервісу, завантаженість каналів зв'язку та ін. Це дозволяє спільно з управлінням по відхиленню використовувати принцип управління, що враховує збурення і реалізувати в системі управління комбіноване управління, при цьому більшу вагу має управління по відхиленню. На рис. 6.9 наведено результат декомпозиції координатора, що реалізує комбінований принцип управління рівнем надання послуг. Координатор на рис. 6.9 містить контур негативного зворотного зв'язку і ланцюги для компенсації збурюючих впливів $\xi \in \Xi$. Компенсаційний контур оцінює основні збурення, які враховуються при виборі коригувальних сигналів.

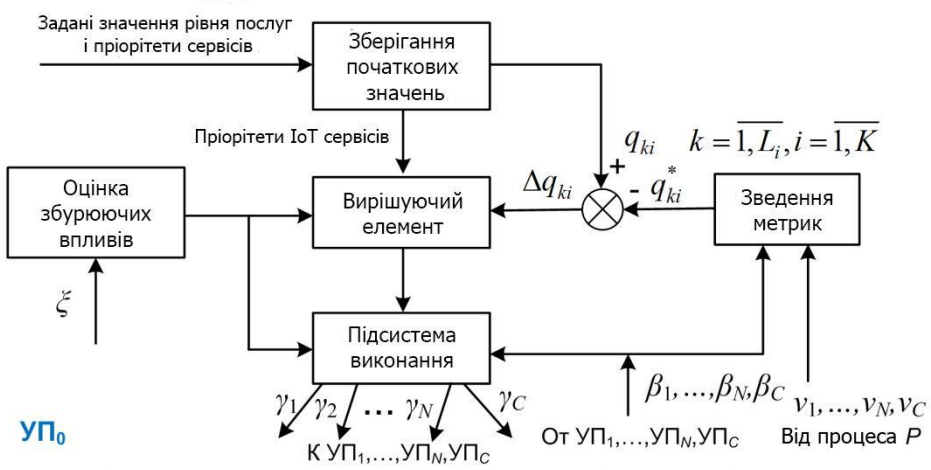


Рис. 6.9. Структура координатора, що використовує зворотний зв'язок і враховує збурюючі впливи

Глобальна мета управління МС-С інфраструктурою може змінюватися, що проявляється для координатора зміною пріоритетів Pr застосунків $\{A_i\}$, що забезпечують надання множині $\mathcal{G}=\{s_i\}$, $i=\overline{1,K}$, послуг IoT і цільових значень $Q_i, i=\overline{1,K}$ рівня послуг. У цьому разі вираз (6.6) виглядає у такий спосіб:

$$C_0 : Pr \times Q_i \times \mathrm{B} \times V \times \Xi \to \Gamma, i=\overline{1,K}. \qquad (6.12)$$

Після визначення принципу управління необхідно визначити стратегію і правила координатора, а також умови застосування стратегії або правил.

Вибір координуючого впливу визначає система

$$< Pr, \Delta\hat{q}, \mathrm{B}, V, \Xi, \mathcal{N} >, \qquad (6.13)$$

де $\Delta\hat{q}$ - вектор відхилення; $\mathcal{N}$ - ситуаційна невизначеність.

Проаналізуємо систему (6.13). Пріоритети Pr задаються застосункам замовником, в інтересах якого функціонує система IoT, і приймають значення з множини $\{1,2,...,Pr_m\}$, де Pr_m - максимальне значення пріоритету. У процесі функціонування МС-С інфраструктури та систем IoT пріоритети застосунків змінюються, відстежуючи зміни значущості бізнес-процесів або процесів діяльності в інтересах яких використовується система IoT.

Для характеристики ступеня відхилення значень елементів вектора $\Delta\hat{q}=(\Delta q_{1,1},...,\Delta q_{k,i},...,\Delta q_{M_K,K})$ від цільових значень введемо функцію $w(\Delta q_{ki})$,

$\forall k = \overline{1, M_i}$, $\forall i = \overline{1, K}$, приймаючу значення на відрізку [-1,1] і яка визначає ступінь близькості фактичного рівня якості до цільових значень

$$w(\Delta q_{k,i}) = (q_{ki} - q_{ki}^{*}) / q_{\text{кр}_{ki}}, \qquad (6.14)$$

де $q_{\text{кр}_{k,i}}$ - критичне значення показника якості *i*-ї послуги, при якому якість вважається незадовільною. Причому при $w(\Delta q_{k,i}) \in (0,1]$ фактичне значення показника якості *i*-ї послуги краще, ніж вимагається, при якості послуги $w(\Delta q_{k,i}) \in [-1,0)$ гірше узгодженого.

З усіх видів невизначеностей найбільш характерною для МС-С інфраструктури є ситуаційна невизначеність $\mathcal{N}$, яка характеризується непередбаченими діями користувачів, непрогнозованістю позаштатних ситуацій, труднощами визначення реакції ресурсів на комбінацію факторів, що впливають. При цьому виникає завдання вироблення координуючих $\gamma = (\gamma_1,...,\gamma_n)$ і керуючих $u = (u_1,...,u_n)$ впливів з урахуванням сигналів зворотного зв'язку $v = (v_1,...,v_n)$ і $\beta = (\beta_1,...,\beta_n)$ при впливі збурень $\xi \in \Xi$ в умови невизначеності $\mathcal{N}$.

Оскільки в аналітичному вигляді визначити відповідне відображення не представляється можливим, то виходом із ситуації є використання ітеративної процедури координації [249, 70], яка передбачає участь всіх процесів, що реалізують управління рівнем послуг, наведених на схемі на рис. 6.7. Застосування ітеративної процедури дозволяє виробити прийнятні координуючі дії з огляду на монотонність функцій залежностей значень показників якості від обсягу ресурсів, а також монотонності впливу ситуаційної невизначеності $\mathcal{N}$ на рівень послуг. Тому для управління якістю послуг, що надаються системою IoT в МС-С інфраструктурі для розкриття невизначеності доцільно використовувати ітеративні процедури управління.

Основна функція координатора полягає в узгодженні діяльності $УП_1,…,УП_N,УП_C$ при генерації ними власних рішень так, щоб підвищити сумарний ефект від їх спільних дій. Тому рішення, що приймаються

координатором, впливають на вибір координуючих, а не керуючих впливів [249]. Для вибору координуючих впливів необхідно визначити принцип координації.

В системі управління координуючі дії вказують на те, якому з $УП_1,\ldots,УП_N,УП_C$ віддається перевага при відновленні якості послуг і які методи доцільно використовувати.

Наприклад, при перевантаженні каналів зв'язку виділення додаткових обчислювальних ресурсів застосункам з множини $\{A_l\}$ не відновить рівень послуг. Тому координатор $УП_0$ повідомляє відповідній підсистемі з $УП_1,\ldots,УП_N,УП_C$, що відповідає за управління потоками в мережі, на необхідність обмеження вихідного трафіку застосунків з $\{A_l\}$, що мають нижчий пріоритет. Таку координацію можна реалізувати, наприклад, через використання заснованого на системі продукцій принципу координації. Правила в нотації Бекуса-Наура мають такий вигляд:

$$
\begin{aligned}
&< система\_правил >::=[< продукція >]\\
&< продукція >::=< умова >\rightarrow< наслідок >\\
&< умова >::=[< проста\_умова >]\\
&< проста\_умова >::=< об'єкт >< атрибут >< предикат >< значення >\\
&< об'єкт >::= УП_1,\ldots,УП_N,\ УП_C\\
&< наслідок >::=[< вказівка > | < формула > | < програма >]\\
&< предикат >::= | \neq | < | > | \leq | \geq\\
&< атрибут >::=[< застосунок > | < пріоритет\_застосунку > | < послуга > |\\
&| < бізнес\_процес > | < важливість\_процесу > | < параметр > | < стан\_системи >]
\end{aligned}
\qquad (6.15)
$$

Застосування заснованого на продукціях принципу координації виправдано у випадках, коли ставиться мета поліпшення якості надання послуг, а не досягнення оптимальних показників функціонування МС-С інфраструктури в умовах недостатності інформації про фактори, що впливають на результати координуючих і керуючих впливів.

Відображення (6.6) може мати дуже складний вид, а для системи (6.13) результуючий взаємозв'язок між координуючими $\gamma=(\gamma_1,\ldots,\gamma_n)$, керуючими $u=(u_1,\ldots,u_n)$ впливами і виходом процесу *P* отримати аналітично неможливо. У цьому разі можна використовувати програмне керування.

Основними процедурами координації є використання ітеративних процедур щодо поліпшення координуючих сигналів на підставі аналізу

результатів координації або використання зворотного зв'язку для корекції координуючого сигналу [249]. Доцільно застосування обох типів процедур. В обох випадках для визначення сигналу помилки при оцінці впливу необхідно провести зведення метрик, вимірюваних на рівні процесу *P*, до метрик, якими оперує координатор.

Необхідно відзначити, що координатор використовується при автоматичному режимі управління рівнем послуг, а при автоматизованому управлінні роль координатора виконує система підтримки прийняття рішень.

Висновки до розділу 6

При розробленні та впровадженні нових систем IoT необхідно враховувати вимоги та відповідні технологічні особливості, щоб існуюча IT-інфраструктура була здатна обробляти великі обсяги даних в реальному часі та адаптуватися до зміни робочих навантажень.

Розроблено підхід до управління інфраструктурою IoT на основі використання мікрохмари для забезпечення бажаної якості IT-послуг з раціональним використанням IT-ресурсів. Ефективність управління IT-інфраструктурою системи інтернету речей пропонується оцінити за якістю послуг та витратами на управління. Запропонований підхід базується на декомпозиційно-компенсаційному підході, в якому завданням оперативного управління якістю послуг є підтримання певного рівня якості обслуговування з використанням мінімального обсягу IT ресурсів. Це дозволяє ефективно використовувати ресурси для надання IT-послуг в системі інтернету речей через виділення рівнів координації послуг, планування ресурсів та управління рівнем обслуговування в інтегрованій системі управління IT-інфраструктурою, а також визначення функцій цих рівнів [279].

Запропонована архітектура IoT на основі MC-C дозволяє ефективно використовувати ресурси для забезпечення IT-послуг в екосистемі IoT. За допомогою реалізації координації рівня обслуговування стає можливим планування ресурсів та управління рівнем надання послуг в інтегрованій системі управління IT-інфраструктурою. Запропонована інформаційна технологія, що

забезпечує розроблення комплексної, інтегрованої IT-інфраструктури в екосистемі IoT. Це дає можливість розробляти нові послуги IoT, що використовуються в певній галузі.

В основі управління IT-інфраструктурою в екосистемі IoT закладена інтегрована взаємодія трьох процесів. Ці процеси включають в себе: координацію на рівні послуг, планування ресурсів і управління рівнем послуг. Процес координації на рівні послуг реалізується ініціативою бізнес-менеджерів і завершується створенням або оновленням наборів послуг та наборів індикаторів якості для кожної послуги. Для реалізації кожної послуги виділяється заздалегідь визначена кількість IT-ресурсів. Процес планування ресурсів базується на розподілі та призначенні IT-ресурсів з пулу ресурсів до кожної з послуг. Процес управління рівнем послуг управляє IT-інфраструктурою таким чином, щоб фактичні значення показників якості послуг відповідали координованим значенням, отриманим у процесі координації рівня обслуговування.

Ядро мікрохмари пропонується реалізувати з використанням гіперконвергентних та програмно-визначених систем, які дозволяють масштабувати інфраструктуру покроково і тільки за необхідності, тим самим збільшуючи доступні ресурси і домагаючись економії, яка зростає з урахуванням розширення IT-інфраструктури провайдера.

Гіперконвергентна інфраструктура використовує не тільки програмно-визначену мережу, а також програмно-визначені системи зберігання даних для модернізації та спрощення середовища ЦОД для реалізації послуг IoT. При цьому, всі політики управління реалізовані на програмному рівні, і вся система не спирається на спеціальне обладнання для виконання основних хмарних функцій, забезпечення робастності та продуктивності. Це дозволяє створювати нову функціональність, яка виконується швидко і якісно, без необхідності модернізації апаратних засобів, мікропрограм або драйверів, що виключає час простою послуг IoT. Інфраструктура мікрохмари IoT може бути налаштована та масштабована в режимі реального часу без необхідності перезавантаження або додаткового придбання обладнання.

РОЗДІЛ 7. ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ УПРАВЛІННЯ ОБЧИСЛЮВАЛЬНИМИ РЕСУРСАМИ IT-ІНФРАСТРУКТУРИ ХМАРНОГО ЦЕНТРУ ОБРОБЛЕННЯ ДАНИХ

У розділі реалізовано інформаційну технологію управління IT-інфраструктурою хмарного ЦОД, розроблено архітектуру і структурну схему ІТУ IT-інфраструктурою хмарного ЦОД, проаналізовано технологічні аспекти та апаратні рішення гіперконвергентної архітектури, обґрунтовано необхідність моделювання IT-інфраструктури за допомогою ArchiMate, розроблено підхід і модель IT-інфраструктури провайдера хмарних послуг, розроблено архітектуру системи управління і моніторингу продуктивності програмно-визначеної IT-інфраструктури хмарного ЦОД.

7.1. Архітектура інформаційної технології

Архітектура інформаційної технології базується на концепції управління IT-інфраструктурою хмарного ЦОД, розробленій у розділі 2, і існуючих поширених практиках розроблення і надання хмарних послуг, використовуваних на практиці. Архітектура інформаційної технології (рис. 7.1) включає в себе систему управління IT-інфраструктурою, до якої надходять управляючі впливи адміністраторів, працюючих в локальній мережі ЦОД, і запити на управління і обслуговування з боку споживачів (користувачів) хмарних послуг, працюючих в мережі Інтернет.

Функції управління хмарними послугами на всіх рівнях сервісної моделі виконуються через портал провайдера після відповідної аутентифікації і авторизації. Провайдер хмарних послуг завжди використовує систему надання хмарних послуг, яка виконує основні функції стосовно створення, розгортання, обслуговування і вивільнення необхідних користувачеві хмарних послуг, задіяючи потрібні для цього обчислювальні ресурси у вигляді ВМ і контейнерів, а також ресурсів локальної мережі ЦОД і сховища.

Рис. 7.1. Архітектура інформаційної технології управління IT-інфраструктурою хмарного ЦОД

Як правило, система надання хмарних послуг створюється за модульним принципом з метою залучення тільки тих модулів управління, які забезпечують виконання певної множини хмарних послуг користувачеві і сервісних (забезпечуючих) послуг, таких як реєстри ресурсів, каталоги користувачів, сервіси сертифікатів, підсистеми обліку використання послуг, сервери імен, сервери управління IP-адресами, сервери БД для зберігання станів підсистем провайдера та ін.

При необхідності розгортання нових хмарних послуг або нових службових сервісів система надання хмарних послуг дозволяє підключити відповідний модуль і за рахунок використання відповідних API інтегрувати його в СУІ [349].

В умовах реального хмарного ЦОД провайдери хмарних послуг використовують три варіанти систем надання хмарних послуг: власні розробки із захищеним вихідним кодом, які не дозволяють вносити зміни в логіку функціонування алгоритмів і методів управління ресурсами, а дозволяють тільки підбирати параметри існуючих алгоритмів і методів (Azure, Google Cloud, AWS); системи з відкритим вихідним кодом і модульною структурою (OpenStack, CloudStack), які дозволяють розробляти і розгортати нові модулі управління,

розширювати функціональність, використовувати нові апаратні засоби і архітектури, такі як гіперконвергентна архітектура і георозподілена архітектура.

Структурна схема інформаційної технології управління IT-інфраструктурою хмарного ЦОД (рис. 7.2) включає в себе модулі управління, які вже впроваджені в систему надання хмарних послуг, і модулі управління, які реалізують інформаційну технологію, запропоновану в дисертаційній роботі.

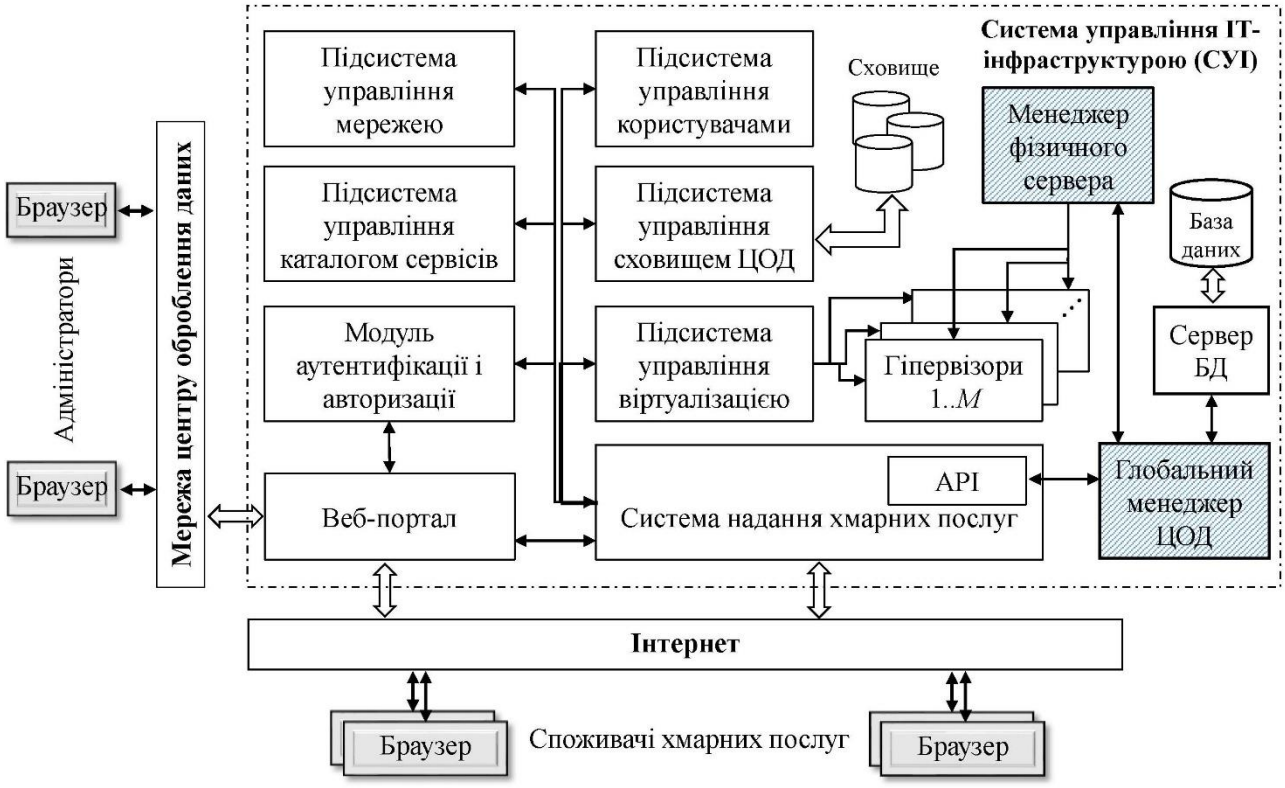


Рис. 7.2. Структурна схема інформаційної технології управління IT-інфраструктурою хмарного ЦОД

Розроблені в розділах 2-7 методологія управління, інформаційна технологія, операторний підхід, стратегії управляння, методи і алгоритми реалізуються у вигляді окремих модулів: глобальний менеджер ЦОД і МФС. Глобальний менеджер ЦОД розгортається і інтегрується на рівні центральної частини системи надання хмарних послуг, а МФС розгортається на кожному ФС, який знаходиться в пулі хмарних ресурсів [349]. Розгортання окремих частин і компонентів інформаційної технології показано на рис. 7.3.

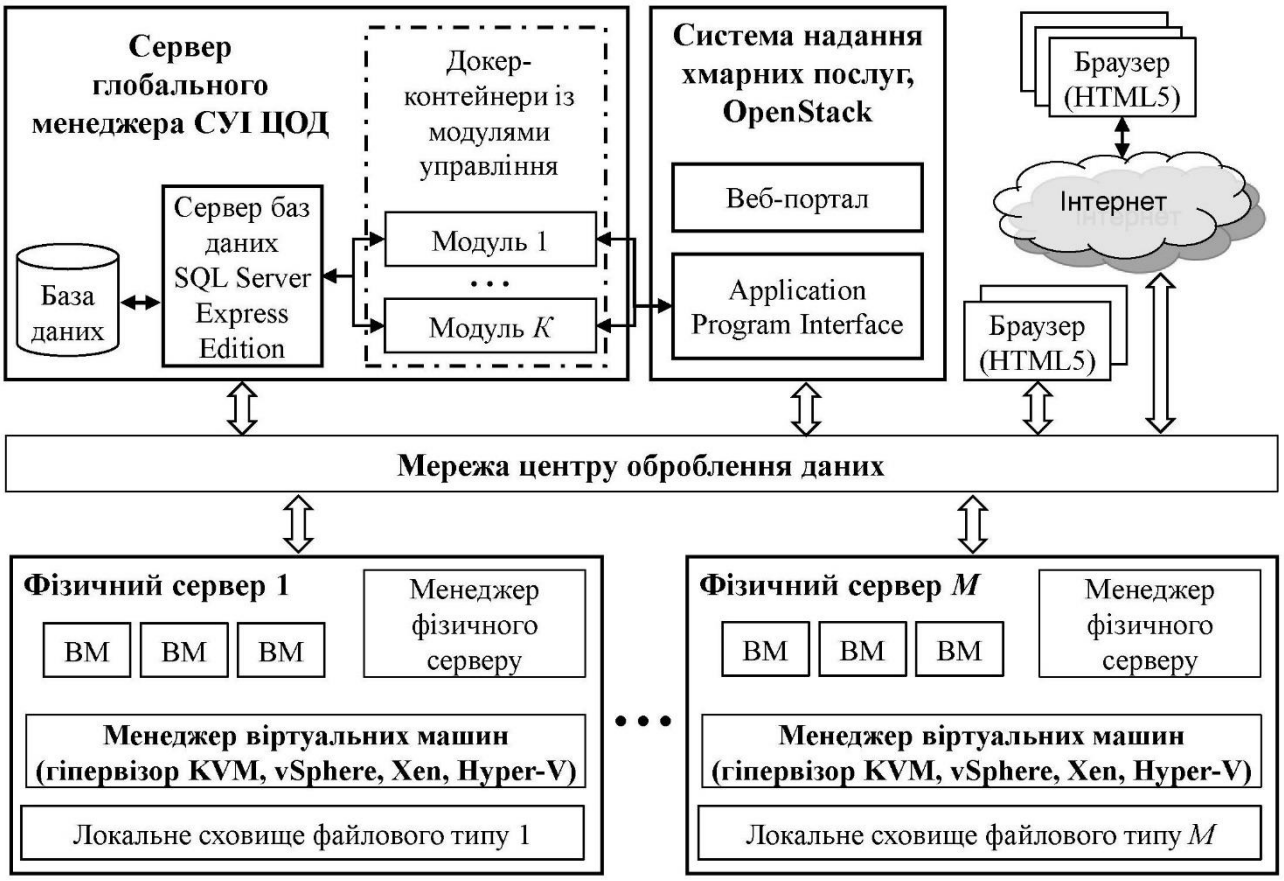


Рис. 7.3. Схема розгортання інформаційної технології управління IT-інфраструктурою хмарного ЦОД

Для виконання функцій глобальний менеджер ЦОД використовує базу даних, в якій зберігає дані моніторингу, поточних параметрів методів і алгоритмів, значення метрик і інші службові дані. У ролі СУБД використано SQL Server Express Edition, так як ця система є безкоштовною, достатньо потужною для виконання поставлених функцій, нескладною в керуванні і розширюваною. Модулі управління, які реалізують розроблені в дисертації стратегії управління, схеми реалізації і відповідні методи (табл. 2.1) реалізовані у вигляді докер-контейнерів, що дозволяє додавати інші оптимізаційні методи управління, ізолювати вплив виконання методів один на одного, використати поширену технологію управління віртуалізацією застосунків, заощадити ресурси сервера управління, на якому розгорнутий глобальний менеджер СУІ. Система надання хмарних послуг реалізована на базі ПЗ OpenStack, яке є дуже поширеним серед провайдерів хмарних послуг в світі, зокрема в Україні, Німеччині та Литві. При цьому модулі управління глобального менеджера ЦОД взаємодіють з модулями системи OpenStack через добре документований API як самої системи, так і її

модулів, що розробляються IT-спільнотою OpenStack. Робота користувачів і адміністраторів з СУІ і хмарними послугами виконується через браузер з підтримкою HTML5. Ця практика стала дуже поширеною при роботі з IT-послугами в глобальному масштабі, так як дозволяє використовувати вся наявні пристрої користувачів, які в змозі надати стандартні елементи інтерфейсу HTML5 (планшети і смартфони). МФС реалізований у вигляді спеціальної ВМ (dom0) і взаємодіє з гіпервізором Xen. Після певної модифікації МФС може бути пристосований для використання з іншими гіпервізорами. Оскільки у функції МФС входить прогнозування навантаження треба зберігати дані моніторингу, отримані з гіпервізора, в локальному файловому сховищі.

Довідки впровадження ІТУ, а також результатів дисертаційної роботи наведені у додатку В.

7.2. Система управління на основі гіперконвергентної архітектури з урахуванням технологічних аспектів

Унаслідок досліджень в області створення високопродуктивних і масштабованих обчислювальних систем, лідерами публічних хмарних сервісів (Microsoft, Google, Amazon) протягом останніх 15-ти років запропонований і випробуваний новий підхід до побудови IT-інфраструктури, який отримав назву веб-масштабування (web-scale). Незважаючи на те, що кожен хмарний сервіс характеризується такими показниками, як використання певної (гомогенної) платформи, дуже велика кількість клієнтів, управління великими обсягами даних (у порівнянні з даними підприємства), ідеї та технології, закладені в підхід веб-масштабування можна застосувати і для гетерогенних платформ в умовах ЦОД підприємства завдяки віртуалізації. Таким чином, ідеї веб-масштабування знайшли своє застосування в гіперконвергентних системах (ГС).

У ГС використовуються принципи і технології розподілених, високопродуктивних і програмно-визначених систем. В основі підходу веб-масштабування знаходяться такі системи і технології, як Cassandra – розподілена система управління NoSQL базами даних; розподілена файлова система HDFS; MapReduce – програмна модель для реалізації розподілених паралельних

обчислень; Hadoop – програмний фреймворк для зберігання і оброблення об'єктів в розподіленому обчислювальному середовищі; Paxos – сімейство протоколів для організації взаємодії вузлів розподіленої системи з метою забезпечення їх узгодженої взаємодії і цілісності даних; Zookeeper – сервіс, який використовується в кластерних розподілених системах для іменування об'єктів і їх синхронізації; REST-based програмний інтерфейс для автоматизації процесів в IT-інфраструктурі.

Як правило, на цей час постачальники ГС пропонують свій програмний стек, в якому використовуються комбінації різних технологій управління і оркестрації, включаючи пропрієтарні і відкриті фреймворки і архітектури. Такий вектор розвитку ГС знижує можливості підприємства і провайдера по інтеграції інструментів і апаратного забезпечення. Таким чином, виникає необхідність в розробленні відкритих архітектур з інструментами оркестрації і інтеграції, заснованих на програмно-визначеному підході і охоплюють всі рівні взаємодії в хмарному ЦОД.

Слід зазначити дуже важливий результат розгортання і застосування ГС в життєвому циклі IT-інфраструктури підприємства, який полягає в одночасному, паралельному використанні обладнання різних поколінь, а також в можливості безшовної інтеграції нових вузлів, що працюють на основі новітніх апаратних архітектур і елементної бази. Таким чином, ГС забезпечує еволюціонування IT-інфраструктури через забезпечення можливості додавання нових ресурсів для вирішення нових завдань і підвищення продуктивності, а також виведення з експлуатації застарілих елементів без впливу на процес надання послуг.

Проаналізувавши спектр програм і сервісів, які використовуються сучасними підприємствами, можна зробити висновок, що повний перехід на використання ГС стримується такими чинниками:

- підприємства використовують різні групи (команди) для адміністрування мережі, систем зберігання даних та віртуалізованих сервісів;
- багато застосунків, які використовуються підприємствами для реалізації бізнес-процесів не спроектовані для розподілених середовищ виконання;

- IT підприємства все ще зосереджені на використанні централізованих сховищ даних;
- низька готовність до застосування нових архітектур розроблення застосунків і виконання операцій управління DevOps.

ГС заснована на віртуалізації мережевих об'єктів і функцій, віртуалізації системи зберігання даних та віртуалізації обчислювальних ресурсів групи ФС. Високий ступінь масштабованості ГС дозволяє нарощувати обчислювальні ресурси і ресурси зберігання даних з необхідним збільшенням, яке обумовлено збільшенням навантаження, поточними та перспективними запитами клієнтів, впровадженням нових сервісів. Лінійне зростання ємності та продуктивності з гарантованою якістю обслуговування є дуже важливою перевагою ГС. При цьому час незапланованих простоїв знижується майже в два рази [141]. Різні IT-команди більше не контролюють відокремлені підсистеми, такі як налаштування, управління та підтримка гіпервізора, сховища та мережі. Ці функції виконуються через API, використовуючи інструменти оркестрації та інтеграції.

На відміну від традиційних архітектур ЦОД, де ВМ зберігалися в загальному сховищі, ГС використовує розподілене сховище, коли кожен вузол системи виконує і функції зберігання, і обчислювальні функції. Гіперконвергентна система з m вузлів показана на рис. 7.4. Кожен вузол може складатися з одного або більше ФС.

Кожен ФС має локальне сховище, складене з NVMe PCI, SSD і HDD дисків, і представлене в системі як програмно-визначене, відмовостійке розподілене сховище з локальним кешуванням на кожну ФС. Реалізація функцій зберігання даних і управління кластером виконується або в модифікованому гіпервізорі (наприклад, Scale Computing), або у вигляді Controller VM (наприклад, Nutanix). Це забезпечує надмірність, високу продуктивність читання даних [142] і відмовостійкість за рахунок розподілу даних по вузлах кластера у вигляді 2-х або 3-х копій одних і тих же даних. Незважаючи на те, що корисний дисковий простір при цьому зменшується, така архітектура дозволяє за-замовченням, також,

забезпечити відмовостійкість у разі відмови одного з трьох вузлів чи виведення вузла з кластера для обслуговування (установка оновлень, профілактика і т.п.).

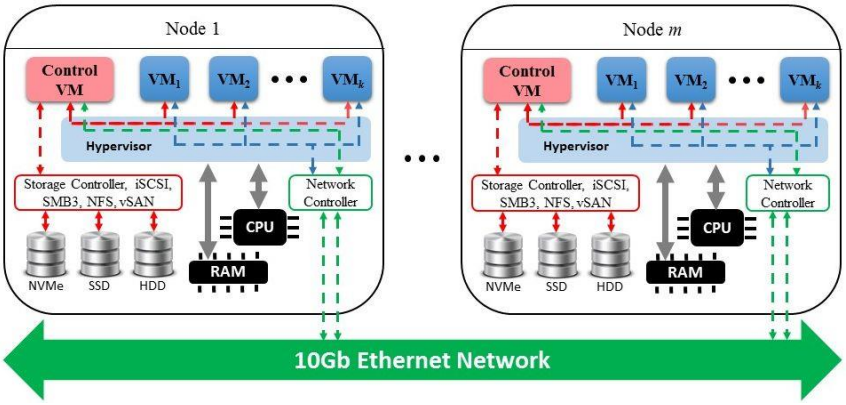


Рис. 7.4. Гіперконвергентна архітектура

Кластер створюється починаючи з 2-х, 3-х або 4-х вузлів, залежно від постачальника гіперконвергентного рішення і стартової функціональності гіперконвергентного кластера. Мережева взаємодія вузлів кластера виконується з використанням двох поширених підходів: з використанням виділеної мережі для функціонування гіперконвергентного кластера (англ. Private/Backplane Network, Out-Of-Bamd) і з використанням єдиної мережевої інфраструктури (англ. Public Network).

7.3. Апаратні рішення для гіперконвергентної системи

Розглянемо кілька апаратних конфігурацій від провідних виробників гіперконвергентних систем. Кожна з представлених конфігурацій характеризується мінімальною і максимальною кількістю вузлів (серверів) в кластері, функціональністю пропрієтарного програмного забезпечення, вимогами до топології мережі, обсягом дискової і оперативної пам'яті одного вузла, кількістю процесорних ядер одного вузла. Прилад ГС може складатися з гомогенних або гетерогенних вузлів. При цьому, вузли або надаються самим виробником, або клієнт може придбати і конфігурувати сервери самостійно. Усі розробники гіперконвергентних систем надають повністю програмно-визначені платформи з підтримкою резервного копіювання, дедуплікації, аварійного відновлення, стиснення і інших інструментів рівня підприємства.

Nutanix є одним з лідерів з розроблення гіперконвергентних систем і пропонує апаратно-програмну платформу на базі обраного користувачем гіпервізора (Acropolis, Hyper-V, ESXi, KVM) для розгортання IT-інфраструктури для будь-яких застосунків (enterprise або web-scale) будь-якого масштабу. Пропоноване рішення дозволяє почати побудову кластера з трьох вузлів і нарощувати продуктивність (scale-out) гнучко, залежно від необхідності, по 20 або 80 ядер на один вузол і 5 ТВ дискового простору. Для управління кластером Nutanix не використовує виділену мережу (Backplane Network) для взаємодії між вузлами, а використовує стандартну телекомунікаційну мережу зі швидкістю передачі 10GbE і 40GbE. Використовується вузол розміром 2U з кількістю ядер 80 [143]. Дослідження IDC [141] в області впровадження ГС Nutanix показують, що щорічні витрати на IT-інфраструктуру знижуються на 30,6%, а загальна вартість володіння (англ. TCO) за п'ятирічний період знижується на 58,3%.

Компанія VMware представила на ринок гіперконвергентних систем кілька рішень: vSAN ReadyNode [144], VCE VxRail, EVO SDDC і EVO: RAIL [144].

vSAN ReadyNode [144] позиціонується як інтелектуальне, економічне та надійне комплексне рішення, яке об'єднує обчислювальні ресурси, засоби віртуалізації, мережі, засоби зберігання даних та управління в пристрій для створення гіперконвергентної інфраструктури. Для розгортання кластера використовуються два вузли, подальше розширення відбувається в 1-вузловому прирості. Кожен вузол містить від 12 до 24 процесорних ядер, до 384 Гб оперативної пам'яті, до 12 ТБ комбінованого жорсткого диска і двох SSD і 10 Гб/с порт для мережного підключення. Один кластер може масштабуватися до 64 вузлів і надавати клієнтові до 120 ВМ на один ФС. Рішення базується на програмних продуктах vCenter Server, vSphere і спеціалізованому програмному забезпеченні.

EVO: RAIL [144] являє собою перший продукт для побудови ГС, анонсований під час VMworld 2014 і позиціонується як інтелектуальне, економічне і надійне комплексне рішення, що об'єднує обчислювальні ресурси, мережу, сховище і засоби управління в пристрій (appliance) для створення

гіперконвергентної інфраструктури. Для створення мінімального кластеру використовуються 4 вузли загальним розміром 2U, подальше нарощування відбувається з кроком в 4 вузли, модулями по 2U. Кожен вузол містить від 6 до 18 ядер процесора, від 128 до 512 ГБ ОП, до 22 ТВ комбінованого сховища HDD і SSD на 4 вузли, два порти 10 Гб/с Ethernet для підключення до мережі і один порт 1 Гб/с Ethernet для мережі управління out-of-band. Застосовується тільки обладнання VMWare з певним формфактором. Один кластер може масштабуватися до 32 серверних вузлів і надавати клієнтові до 2000 ВМ у вигляді робочих столів. Рішення базується на програмних продуктах vCenter Server, Virtual SAN, vSphere Enterprise Plus і спеціалізованого ПЗ для EVO: RAIL.

VCE VxRail [144] повністю інтегрована, попередньо встановлена, протестована і налаштована ГС, яка базується на ПЗ і обладнанні VMware vSphere, VMware Virtual SAN і ПЗ EMC та призначена для побудови програмно-визначеного ЦОД (SDDC). Створення кластера можна почати з одного чотирьох-серверного модуля 2U і масштабувати його до 64 вузлів з кроком в 4 вузли. Один вузол може бути налаштований з використанням обладнання різної продуктивності: процесор або процесори з числом ядер від 6 до 20, ОП від 64 ГБ до 512 ГБ, SSD сховище від 3 ТБ до 6 ТБ, 2 порти 10 Гб/с Ethernet, 2 порти 1 Гб/с Ethernet і один порт для управління модулем out-of-band. Дана система використовує пропрієтарне ПЗ. Загалом це рішення є конвергентним, так як не відповідає вимогам web-scale.

VMware EVO SDDC [144] дозволяє створити приватну хмару за допомогою широкого діапазону рекомендованого апаратного і програмного забезпечення, включаючи сервери і комутатори, які виробляють VMware і її партнери. Початкова конфігурація кластера складається з однієї стійки (Rack) і містить 8 вузлів. Далі, кластер можна масштабувати до 10-ти стійок з кроком в один сервер і отримати 240 вузлів в кластері. Власне ПЗ (VMware vSphere, NSX, Virtual SAN, EVO SDDC Manager) дозволяє виконувати оркестрацію приватної хмари з однієї точки входу і програмно-конфігурувати всі складові хмарного рішення: обчислення, сховище та мережеву взаємодію. Таким чином, дана платформа реалізує сервіси IaaS і VDI. Логіка управління сховищем інтегрована в гіпервізор

vSphere, є можливість отримувати доступ до зовнішніх сховищ з використанням протоколу IP. Для роботи однієї стійки необхідно використовувати два ToR комутатора з портами 10 Гб/с Ethernet, до яких одночасно підключається кожен сервер, а також керуючий комутатор 1 Гб/с Ethernet. Ядро кластера створюється з двох комутаторів Inter Rack Spine Switch 40 Гб/с Ethernet, до яких одночасно підключається кожен ToR комутатор.

Dell EMC VxRail [145] є повністю інтегрованою, попередньо встановленою, перевіреною та налаштованою ГС, яка базується на програмному забезпеченні VMware vSphere та обладнанні Dell EMC. Пристрій VxRail із інтеграцією рівня ядра між VMware vSAN і гіпервізором vSphere. VxRail розроблений для створення програмно-визначеного ЦОД. Розгортання кластера може бути запущено з трьох вузлів, і його можна масштабувати до 64 вузлів з кроком в один вузол. Один вузол може бути налаштований з використанням різних апаратних засобів: процесор(и) з числом ядер до 44, оперативна пам'ять від 64 ГБ до 512 ГБ, накопичувач SSD від 3 ТБ до 6 ТБ, два порти 10 Гб/с і два порти 1 Гб/с.

Платформа віртуалізації HC3 від Scale Computing позиціонується як «datacenter in a box» і дозволяє почати побудову кластера з використанням трьох вузлів з однаковою апаратною конфігурацією в межах одного з рівнів (HC1000, HC2000, HC5000). Максимальне число вузлів в кластері – 8, проте можливо адаптувати кластер для роботи з великою кількістю вузлів з залученням консультантів Scale Computing. HC3 спроектована без єдиної точки відмови, для управління кластером використовується Private/Backplane Network (1 Гб/с Ethernet або 10 Гб/с Ethernet). Кожен вузол системи HC3 має розмір 1U [146].

7.4. Потреба моделювання IT-інфраструктури

Сучасний комплекс програмно-апаратних засобів реалізації бізнес-процесів корпорації характеризується зростанням складності, динамікою технологічних змін, збільшенням обсягів оброблюваної інформації. В основі реалізації бізнес-процесів корпорації лежить ефективна IT-інфраструктура [296]. Від ефективності побудови та організації IT-інфраструктури залежить реалізація місії корпорації, а значить конкурентоспроможність і можливість швидко

реагувати на зміни ринку. У цих умовах IT-інфраструктура робить вирішальний вплив на якісні показники роботи прикладних сервісів, такі як час відгуку системи, доступність, продуктивність і безпеку [321].

При управлінні IT-інфраструктурою необхідно використовувати засоби моделювання, впровадження, розгортання і підтримки. Початковим етапом аналізу IT-інфраструктури є моделювання. Системними архітекторами широко використовуються як пропрієтарні, так і відкриті мови і фреймворки: ArchiMate, Systems Modeling Language, Zachman Framework, UML та інші. Мова моделювання архітектур підприємства ArchiMate є відкритим і незалежним від вендора засобом моделювання, що забезпечує нотацію, яка дозволяє корпоративним архітекторам подати опис, аналізувати і візуалізувати відносини між елементами ділових доменів єдино можливим чином [235].

Моделювання IT-інфраструктури за допомогою ArchiMate дозволяє:

- проаналізувати роботу технологічного рівня, на якому будуть розгорнуті застосунки;
- визначати ділянки IT-інфраструктури, які потребують оптимізації;
- проводити аудит і визначати відповідність вимогам;
- підвищити ефективність управління змінами IT-інфраструктури;
- візуалізувати елементи IT-інфраструктури, на яких буде функціонувати той чи інший застосунок;
- визначити рівень доступу і політику безпеки при доступі до елементів IT-інфраструктури;
- підвищити ефективність пошуку несправностей і знаходження вузьких місць в продуктивності.

Унаслідок аналізу існуючих архітектур і прикладів впроваджень можна виділити такі основні властивості IT-інфраструктури [303]:

- надає послуги і застосунки, які виконуються з її використанням;
- використовується кількома застосунками або сервісами одночасно;
- IT-інфраструктура є більш статичною і менш схильна до змін, ніж застосунки і програмні засоби;

- зазвичай знаходиться під управлінням персоналу, який не керує рівнем сервісів і програмними засобами [236];
- зміни в IT-інфраструктурі обумовлені необхідністю змін на рівні застосунків і бізнес-процесів.

Розвиток корпорації, освоєння нових ринків і нових видів послуг призводить до необхідності управління IT-інфраструктурою в нових умовах [292]. Ці фактори назвемо якісними. Необхідність управління IT-інфраструктурою може бути викликана і кількісними факторами, таким як зміна обсягів оброблюваної інформації, зміна чисельності працівників і підключених до мережі робочих місць, зміна чисельності клієнтів компанії. Причому, якщо відбувається збільшення кількісних показників, то необхідно змінювати управління IT-інфраструктурою в напрямку підвищення продуктивності і скорочення витрат TCO. Якщо відбувається зменшення кількісних показників, то зміни будуть спрямовані на скорочення витрат TCO і зниження енергоспоживання.

Управління IT-інфраструктурою, як системою, полягає у визначенні такого її кінцевого стану, при якому зміни її структури, якісних і кількісних характеристик її підсистем призводять до досягнення максимуму критерію якості надання послуг і мінімуму витрат на виконання операцій управління. У складі критерію якості враховують показники угоди про надання сервісу SLA організації, TCO, ROI та час відгуку системи на дії кінцевого користувача. Важливим, також, є питання про необхідність оптимізації IT-інфраструктури. Крім того, на прийняття рішення про необхідність оптимізації IT-інфраструктури впливають організаційні причини, виміряти які, найчастіше, не представляється можливим. На верхньому рівні абстракції процес управління IT-інфраструктурою розглядається як визначення керуючих впливів з метою її переходу від поточного стану до бажаного, що супроводжується поліпшенням якісних показників роботи інформаційних сервісів.

7.5. Елементи мови ArchiMate для моделювання IT-інфраструктури

IT-інфраструктура відноситься до самого нижнього рівня моделі архітектури підприємства, яку надає мова ArchiMate (рис. 7.5).

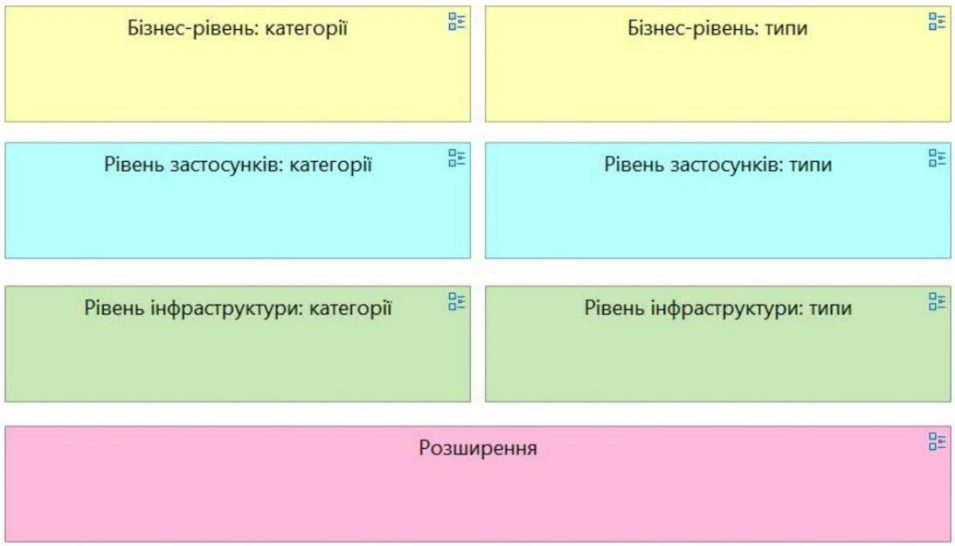


Рис. 7.5. Рівні моделі архітектури підприємства в ArchiMate [342]

Рівень технологій надає послуги як *рівню застосунків*, так і деяким споживачам усередині самого рівня. На практиці цей рівень має більш конкретну назву – рівень IT, або рівень IT-інфраструктури. Елементи цього рівня позначаються на моделях зеленим кольором.

Основним елементом моделі рівня IT-інфраструктури є *вузол*. Вузол є моделлю апаратного забезпечення. Поведінку вузла в цілому відображає *функція*, видиму поведінку якої визначено в моделі за допомогою *послуги*. Вузол реалізовує свою послугу для рівня застосунків через *інтерфейс*.

Елементи моделі дозволяють представляти IT-інфраструктуру з різних точок зору [342], серед яких для моделювання на практиці застосовуються Infrastructure Viewpoint, Infrastructure Usage Viewpoint і Implementation and Deployment Viewpoint. Клас моделей Infrastructure Viewpoint застосовується при розробленні моделей нових IT-інфраструктур. Infrastructure Usage Viewpoint і Implementation and Deployment Viewpoint необхідні при моделюванні процесів використання, реалізації та розгортання IT-інфраструктури. Для моделювання архітектури IT-інфраструктури використовуються такі елементи мови ArchiMate (рис. 7.6).

Рис. 7.6. Елементи моделювання IT-інфраструктури ЦОД [342]

Архітектуру IT-інфраструктури можна подати узагальнено або детально. Вибір рівня деталізації є завданням системного архітектора [237]. За допомогою елементів рівня ядра ArchiMate (Core elements) не завжди можливо створити модель мережевих пристроїв і їх взаємодії. Крім того, одну і ту ж мережеву архітектуру можна представити різними наборами елементів і зв'язків. Наприклад, для позначення мережі мова пропонує два елементи: Network і Communication Path. Деякі інструменти моделювання надають, також, додаткові графічні примітиви для позначення мережевих пристроїв.

7.6. Підхід до моделювання IT-інфраструктури

На сьогодні інформаційні технології надаються провайдерами хмарних IT-послуг згідно трьох основних моделей: SaaS, PaaS і IaaS [272]. Розглянемо загальний випадок моделювання IT-інфраструктури за моделлю IaaS.

Для побудови ядра IT-інфраструктури використовуються три основних компоненти фізичного рівня: серверні системи, сховища і мережеві пристрої. Пропонується для позначення цих компонентів використовувати елемент мови вузол (рис. 7.7). Для IT-інфраструктур різної складності і призначення число вузлів N ($2<N<K$, де K – максимальна кількість вузлів, обмежена можливостями ЦОД, такими як електроживлення, розміщення і охолодження) розраховується

виходячи з вимог SLA, кількості клієнтів *M*, кількості прикладних сервісів і сервісів IT-інфраструктури.

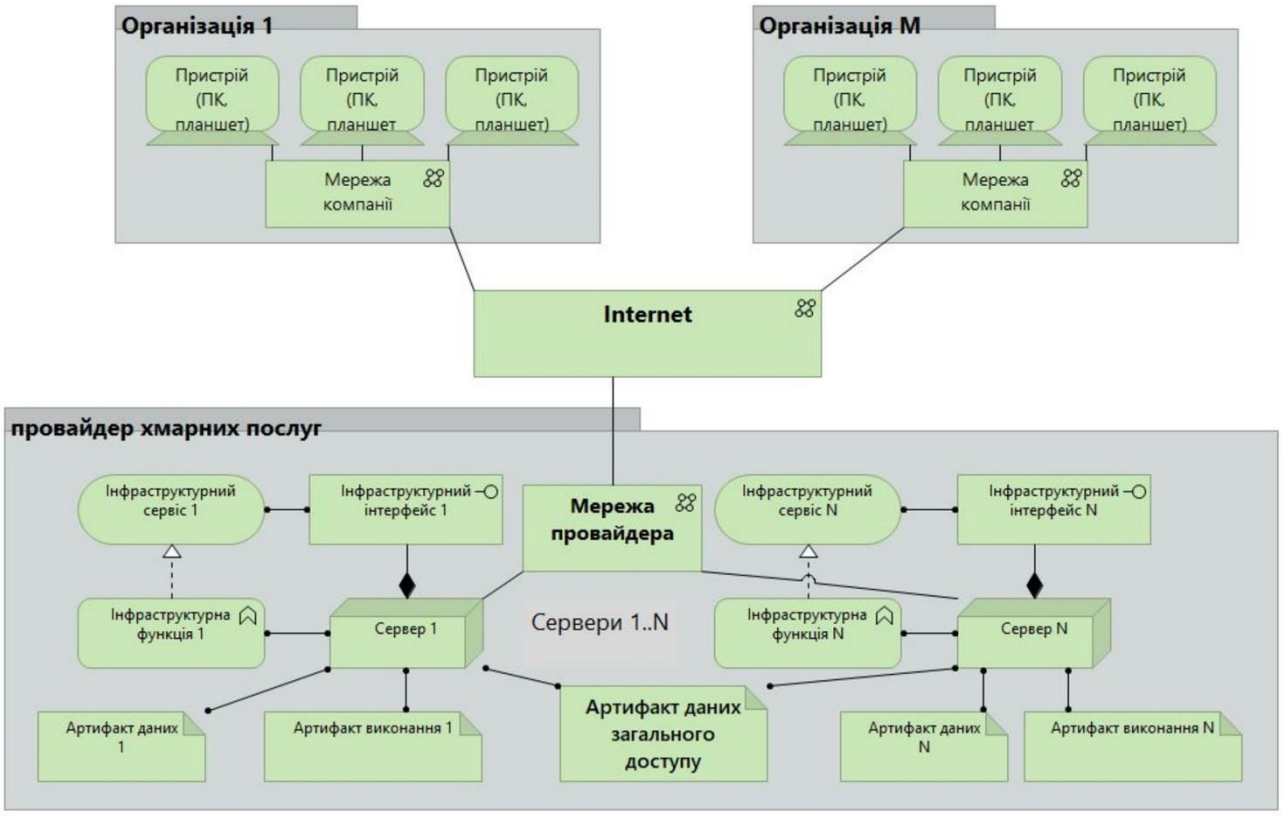


Рис. 7.7. Модель інфраструктурного рівня провайдера хмарних послуг

Кожен вузол має такі властивості: продуктивність, число портів, тип портів (інтерфейс). Використовуючи запропоновані властивості вузла, архітектор має можливість побудувати модель IT-інфраструктури фізичного рівня в термінах мови ArchiMate. Отримані моделі фізичної структури IT-інфраструктури мають однорівневу архітектуру без ієрархій. Це пов'язано з тим, що завдяки збільшенню пропускної здатності мереж передачі даних, введення в експлуатацію нових ЦОД і застосування конвергентних технологій немає необхідності до побудови ієрархічних і розподілених фізичних структур, які складніше проектувати і адмініструвати. Нині спостерігається перехід до централізованих однорівневих фізичних структур.

На діаграмах Infrastructure Usage Viewpoint і Implementation and Deployment Viewpoint вказують такі елементи управління IT-інфраструктурою: Monitoring, Infrastructure Management, Infrastructure Services, Process Management [236]. Ці елементи необхідні і використовуються в будь-якій розвиненій IT-

інфраструктурі. Моніторинг застосовується для спостереження за критичними параметрами IT-інфраструктури з метою дотримання умов SLA, виявлення несправностей і вузьких місць, а також попередження їх виникнення. Система управління інфраструктурою дозволяє в автоматизованому режимі управляти сервісами, запуском і зупинкою систем, установкою оновлень і виконувати інші сервісні функції. Інфраструктурні сервіси виконують додаткові функції для рівня застосунків, пов'язані з веденням журналів, підтримкою каталогів і сервісів іменування, управління допуском і безпеки. Управління процесами здійснюється за допомогою методології управління інфраструктурою, наприклад, ITIL, MOF, COBIT та ін.

7.7. Модульна структура системи управління і моніторингу продуктивності програмно-визначеної IT-інфраструктури

При управлінні складними розподіленими обчислювальними системами за критерієм максимізації продуктивності клієнтських додатків необхідно мати оперативну інформацію про поточну навантаження на елементи IT-інфраструктури. Нині існує тенденція до стандартизації апаратних і операційних платформ IT-інфраструктури, в тому числі поширення програмно-визначених елементів [272]. Результатом стандартизації є використання широкого спектру компонентів для серверів, систем зберігання даних та мережевого обладнання від невеликого числа вендорів [273]. Крім того, кожен вендор пропонує комплекс системного ПЗ для управління та моніторингу, в якому недостатня увага приділена інтеграції з обладнанням інших вендорів [267]. Часто, це системне ПЗ не має API або не підтримує новітню функціональність обладнання інших вендорів, що значно впливає на продуктивність. Особливо важливим є підтримання якості роботи мультимедійних сервісів в мережах провайдерів [268]. Таким чином, завдання збору та аналізу даних про поточний стан елементів IT-інфраструктури з урахуванням програмно-визначених мереж, сховищ і обчислень в гетерогенному середовищі є актуальним завданням.

У сучасних ЦОД функціонують багато різних систем моніторингу [238], слабо інтегрованих, що не дозволяє аналізувати інциденти і виробляти рекомендації, так як спостереження проводиться тільки за певною ділянкою і результати спостереження передаються прямо на рівень прийняття рішень, а не в систему попередньої аналітичної обробки і аналізу. Для інтеграції цих систем підприємства витрачають великі кошти з числа операційних витрат.

На сьогодні весь спектр рішень можна умовно розбити на дві групи: окремі системи моніторингу та інтегровані засоби моніторингу в складі великих систем управління обладнанням певного вендора. До окремих систем і засобів моніторингу можна віднести: утиліти операційних систем (ping, pathping, tracert, nslookup, netsh, netstat і інші); вбудовані програми і засоби операційних систем (журнали подій, лог-файли, Network Monitor, Performance Monitor, ESXTOP, syslog та інші); самостійні розробки, комерційні та безкоштовні системи моніторингу та управління мережею (Nagios, PRTG, MRTG, Zabbix, Zenoss, libpcap / WinPcap, GroundWork, Darkstat і інші).

Серед систем управління мережею з інтегрованими засобами моніторингу, аудиту та журналізації необхідно відзначити Fluke Networks Visual Performance Manager, HP OpenView, EMC Storage Resource Management, IBM Tivoli, Network Forensics, Microsoft System Center, BlueStripe FactFinder, NetApp Akorri BalancePoint, VMware vCenter Operations.

Як об'єкт моніторингу виступають такі компоненти IT-інфраструктури: мережеве обладнання, серверне обладнання, системи зберігання, системи електроживлення, лінії зв'язку (мережі зв'язку), офісне та касове обладнання (принтери, факси, сканери та ін.). Різні системи моніторингу використовують власні терміни для позначення об'єктів моніторингу і їх параметрів. Іноді для позначення операцій моніторингу використовують термін спостереження.

Для будь-якого об'єкта моніторингу рівня підприємства можна виконати декомпозицію на множину об'єктів, представлених спостерігачеві множиною властивостей (параметрів, що контролюються). Значення властивостей об'єкта в

заданий момент часу характеризують його стан. При цьому стан системи в цілому є функцією станів множини підсистем нижчого рівня. Контрольовані параметри мають дві важливі характеристики: актуальність та повнота. Таким чином, при виконанні декомпозиції враховують такі вимоги до спостережуваних характеристик об'єктів моніторингу: достатність, атомарність, незалежність, актуальність [294].

При управлінні IT-інфраструктурою використовуються дані спостереження за процесорними ресурсами, оперативною пам'яттю, мережевою підсистемою і системою збереження даних. Отримання даних про стан завантаження процесорів і оперативної пам'яті не представляє особливих труднощів. Ці дані використовуються при плануванні розміщення ВМ на певних ФС, а також при прийнятті рішення про міграцію ВМ згідно з різними критеріями, безпосередньо пов'язаним з SLA та операційними витратами. Отримання і аналіз даних про стан мережевої підсистеми і про стан сховища є більш складні завданням, при вирішенні яких найчастіше використовуються евристики, шаблони і політики.

Найбільш критичним елементом, який впливає на продуктивність роботи застосунків, на якість обслуговування клієнтів є система збереження даних. Пропускна здатність мережі і кількість пристроїв не є показником продуктивності і поточного завантаження. На продуктивність роботи СЗД впливають такі показники: метрики сховища (IOPS, Throughput, Latency і ін.); фізичні характеристики дисків в масиві (Average rotational latency, IOPS per disk і ін.); тип, швидкість і число мережевих портів, архітектура СЗД з використанням КЕШ і балансування навантаження; продуктивність роботи серверної системи, що використовує СЗД.

Для підвищення ефективності роботи IT-інфраструктури як системи в цілому пропонується такий високорівневий алгоритм. Аналіз роботи і налаштування ФС (процесор, ОП) у складі системи для досягнення показників продуктивності при роботі з поточним навантаженням. Аналіз і приведення у відповідність з поточним і прогнозованим навантаженням мережевої

підсистеми. Спостереження за метриками СЗД з метою перерозподілу навантаження в короткостроковій і середньостроковій перспективі. Прогнозування навантаження на всі три підсистеми і перехід до аналізу роботи ФС. Кожна стадія алгоритму супроводжується оптимізацією визначених параметрів якості.

Розроблений алгоритм пропонується реалізувати в рамках архітектури системи управління і моніторингу програмно-визначеною IT-інфраструктурою, модель якої показана на рис. 7.8. Нині логіка і алгоритми роботи апаратних засобів винесені з рівня даних на програмний рівень. Це дозволило адаптувати роботу пристроїв фізичного і канального рівнів відповідно до потреб сучасних бізнес-процесів. До компоненту АПВМ надходять дані з контролерів програмно-визначених компонентів IT-інфраструктури для планування розміщення ресурсів та їх перерозподілу в разі потреби. АПВМ, також, виконує оцінку метрик, які надійшли з контролерів, і планує розміщення ресурсів. Модуль управління допуском до сервісів управляє ідентифікацією користувачів, балансуванням навантаження через перенаправлення запитів, тарифікацією наданих послуг.

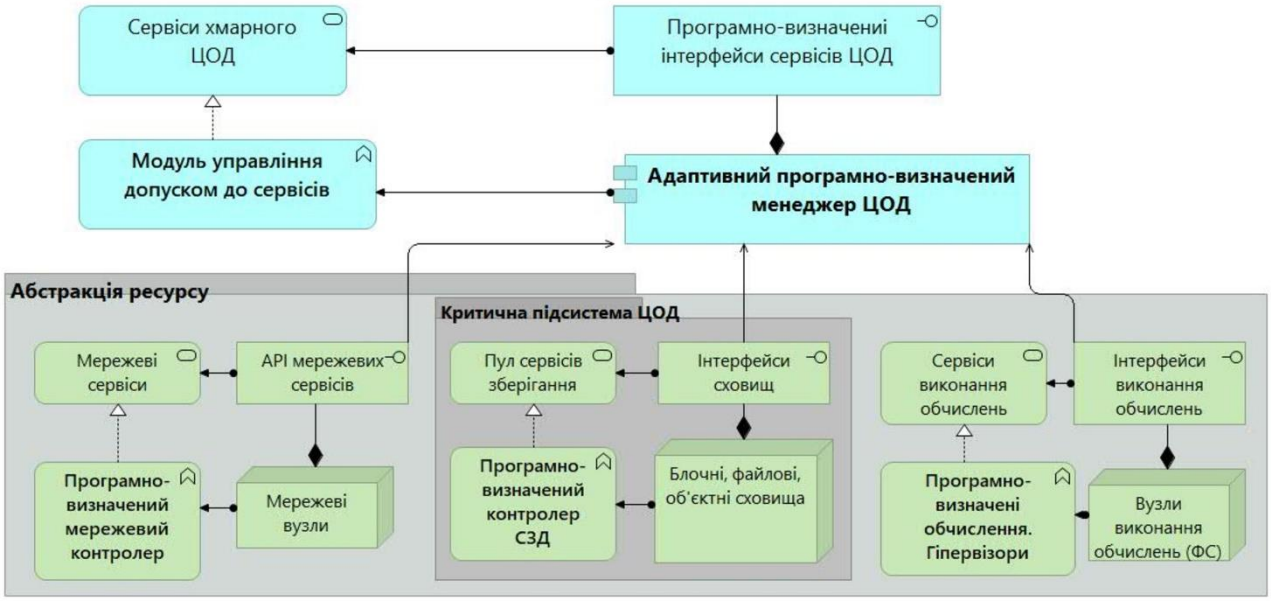


Рис. 7.8. Модель системи управління і моніторингу програмно-визначеною IT-інфраструктурою

Висновки до розділу 7

1. Розроблена архітектура інформаційної технології управління IT-інфраструктурою хмарного ЦОД, а також її структурна схема і схема розгортання. Інформаційна технологія базується на методології управління IT-інфраструктурою хмарного ЦОД, розробленій в п.2.3., і використовує систему управління хмарними послугами з відкритим вихідним кодом і модульною структурою OpenStack, що дозволяє впровадити запропоновані в дисертації підходи, методи і алгоритми у вигляді окремих модулів і інтегрувати їх в існуючих умовах роботи провайдера хмарних послуг. Для розгортання модулів з методами управління на боці СУІ використовуються докер-контейнери. Для розгортання МФС використовується спеціальна ВМ (dom0), яка взаємодіє з гіпервізором Xen.

2. Сучасна IT-інфраструктура ЦОД характеризується значною складністю, а також від ефективності її побудови залежить реалізація місії компанії. Моделювання IT-інфраструктури з допомогою ArchiMate дозволяє підвищити ефективність процесів проектування, розроблення і аналізу складних систем. Моделювання IT-інфраструктури з допомогою ArchiMate дозволяє розробляти моделі для провайдерів послуг SaaS, PaaS і IaaS.

3. Для побудови ядра IT-інфраструктури на практиці використовуються три основних компоненти фізичного рівня: ФС, системи збереження даних і мережеві пристрої, для позначення яких пропонується використовувати елемент мови вузол. Моделі фізичної структури IT-інфраструктури мають однорівневу архітектуру без ієрархії, так як на цей час спостерігається перехід до централізованої архітектури, такої як ЦОД.

4. Програмно-визначені елементи IT-інфраструктури дозволяють уніфікувати засоби моніторингу та більш ефективно управляти процесами виконання застосунків і роботою ВМ. Використовувані на цей час системи моніторингу складно інтегрувати, їх функціональність залежить від архітектури пристроїв і систем конкретного вендора. Найбільш критичним елементом, який впливає на продуктивність роботи застосунків клієнтів і серверного ПЗ є сховище. Для підвищення ефективності управління плануванням розміщення застосунків і ВМ запропонована модель системи моніторингу програмно-

визначеної ІТ-інфраструктури, а також високорівневий алгоритм управління нею.

5. Нині для впровадження ГС в ЦОД провайдера хмарних послуг необхідно адаптувати існуючі програмні системи і сервіси для реалізації можливості розподіленого оброблення даних. Деякі виробники ГС застосовують архітектуру, яка забезпечує роботу звичайних гіпервізорів і існуючих застосунків, тим самим абстрагуючись від реалізації розподіленого оброблення файлів і даних. Незважаючи на це, все ще існує група ГС, для якої необхідно розробляти рівні абстракції на рівні сервісу, інтегровані з вже існуючими сервісами з метою забезпечення роботи застосунків не тільки орієнтованих на розподілену обробку даних, а й вже існуючих застосунків провайдера хмарних послуг.

6. Унаслідок аналізу архітектур, що використовуються в поширених ГС, можна зробити висновок, що багато виробників гіперконвергентних систем використовують пропрієтарне обладнання і ПЗ, що призводить до сильної прив'язки до вендору. Таким чином, інтеграція систем від різних виробників не передбачається, за винятком використання додаткового ПЗ з використанням API, базованого на REST. Незважаючи на те, що на даному етапі ГС в основному позиціонуються для використання інфраструктури ВРС, аналіз архітектур показує, що представлені кластери можна використовувати для будь-яких застосунків корпорації, що також може супроводжуватись розгортанням контейнерів. Питання розгортання приватної хмарної ІТ-інфраструктури вирішені для невеликих кластерів в складі гіперконвергентного рішення (від декількох десятків до декількох сотень ФС). Для великих кластерів, які налічують тисячі ФС проблеми побудови ГС, проблеми реалізації ефективної мережевої взаємодії вузлів і організації системи розподіленого зберігання даних знаходиться в стадії вирішення.

7. У розділі не розглянуто аспект, що враховує розміщення обладнання, формфактори і питання енергозбереження, однак вендори багато зусиль приділяють цьому питанню. Запропоновано використовувати дворівневу координаційну схему для керування обчислювальними, накопичувальними, мережевими та віртуальними підсистемами гіперконвергентної інфраструктури разом з алгоритмами управління в цих підсистемах.

РОЗДІЛ 8. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РОЗРОБЛЕНОЇ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

У розділі представлено результати експериментального дослідження запропонованих у дисертації схем реалізації, моделей і методів та розробленої на їх основі ІТУ. Виконаний аналіз використання моделей динаміки та інтегрованого управління ІТ-інфраструктурою хмарного ЦОД, а також аналіз результатів роботи схем реалізації стратегій управління ресурсами ІТ-інфраструктури, а саме: методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу, двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку, методу динамічної консолідації і розміщення ВМ на основі алгоритму навчання з підкріпленням. Представлені результати експериментальних досліджень адаптивного методу комбінованого прогнозування з різними евристиками комбінування прогнозів, виконана оцінка продуктивності їх роботи. Представлені результати моделювання управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД.

8.1. Результати використання моделей інтегрованого управління ІТ-інфраструктурою

Модель динаміки ЦОД реалізована за допомогою мови C#. При аналізі та моделюванні використані дані з журналів роботи кластера Google [106] та з набору даних Bitbrains [22]. Це обумовлено тим, що штучно згенеровані дані про стан використання ресурсів ЦОД не відповідають реальним умовам експлуатації ЦОД, і при моделюванні перевага віддається актуальним даним, що отримані в умовах виробництва.

Журнали статистики використання кластера Google складаються з декількох наборів да-них про завдання, виконані приблизно на 12000 ФС. Журнали містять записи про конфігурацію ФС, події планування завдань, потреби в ресурсах для завдань та записи про використання ресурсів 25462157 завданнями протягом 29 днів. Для кожного завдання користувач вказує максимальну потребу в кожному ресурсі, насамперед потребу в ресурсах

процесора і розмірі пам'яті. Значення попиту та використання для кожного типу ресурсу нормалізовані від 0 до 1 відносно найбільшої ресурсоємності ФС.

Набір даних Bitbrains містить показники ефективності 1750 ВМ із розподіленого ЦОД, який спеціалізується на постачанні послуг хостингу та бізнес-розрахунків для підприємств. Кожен файл набору даних містить показники роботи однієї ВМ. Як вхідні дані моделі використано один з наборів, fastStorage, який складається з 1250 ВМ, підключених до мережі зберігання даних. У кожному файлі журналу роботи ВМ зберігаються такі важливі дані про продуктивність роботи, як: час початку роботи ВМ, час закінчення роботи ВМ, кількість віртуальних процесорних ядер, запитаний і використаний ресурс процесора ФС, запитаний і використаний ресурс пам’яті ФС, операції з диском (читання і запис даних, КБ/с), операції з мережевим адаптером (отримано і надіслано даних, КБ/с).

У результаті аналізу трейсів роботи кластера Google можна побачити, що попит на кожний тип ресурсу коливається в часі. Це відображається на змінній $c_j^k(t)$ при моделюванні, коли для кожного завдання запланована окрема ВМ. Модель отримує запити на виконання завдань із трейсу роботи кластера Google. Експерименти проводились протягом різних періодів часу з тривалістю 7 днів. Для кожного завдання інформація про попит ресурсів, час появи та час закінчення використана в моделі за рахунок змінних $r_{ij}^k(t)$, $a_{ij}^h(t)$ та $s_{ij}(t)$ (розділ 3). Процес прийняття рішень для визначення нового розміщення ВМ на кожному кроці t виконується за рахунок використання змінних управління $u_{ij}(t)$, $x_{ij}(t)$, $y_i(t)$, $S_i(t)$ (розділ 3), що подаються на вхід алгоритму, який запропонований в [308]. Процес прийняття рішень для вимкнення недовантажених ФС через консолідацію ВМ на кожному кроці t виконується за рахунок використання змінних управління $u_{ij}(t)$, $x_{ij}(t)$, $y_i(t)$ (розділ 3).

Ефективність використання електроенергії оцінюється через порівняння енергоспоживання активних ФС під час моделювання згідно запропонованих моделей з результатами, отриманими за допомогою алгоритму Power Aware Best Fit Decrease (PABFD) [122]. У PABFD всі ВМ сортуються в порядку зменшення

їх поточної утилізації процесора. Розподіл кожної ВМ на ФС у PABFD виконується з урахуванням найменшого збільшення споживання електроенергії, викликаного розподілом. Запропонований в дисертаційній роботі підхід дозволяє розподіляти ВМ на ФС в середньому на 17% ефективніше, ніж PABFD.

У [112] зазначається, що у трейсі Google більше 50% завдань у групі з пріоритетом «продуктив» (production priority) заплановані негайно (за рахунок методів самого Google). Але деякі задачі у пріоритетній групі "продуктив" затримуються протягом декількох годин і навіть днів. У результаті аналізу встановлено, що однією з можливих причин такої ситуації є відсутність належних ресурсів з точки зору відповідної конфігурації ФС. Щоб зменшити до мінімуму таку розбіжність між запитаними та доступними ресурсами, запропоновано використовувати метод управління потужністю. Цей метод дозволяє зарезервувати ФС кожного типу після наступного рішення щодо планування та розгортання нових ВМ. Після застосування запропонованої моделі управління потужністю в сценаріях моделювання, затримка планування зменшується до 37% у деяких сценаріях (вибірках з трейсу). Таким чином, цей метод орієнтований на планування виконання неоднорідних завдань у гетерогенних середовищах.

Процес прийняття рішень для вимкнення недовантажених ФС через консолідацію ВМ на кожному кроці t за рахунок використання змінних управління $u_{ij}(t)$, $x_{ij}(t)$, $y_i(t)$ виконується з урахуванням миттєвого та середнього коефіцієнтів життєздатності ВМ, V_{VM}^{inst} і V_{VM}^{mid} згідно (3.23). Моделювання енергоспоживання ЦОД виконується відповідно до (3.14) з урахуванням показників споживання електроенергії сучасними ФС [123, 126].

8.2. Оцінка результатів моделювання методу рівномірної консолідації віртуальних машин з використанням ідеї імітації відпалу

Для проведення повторюваних експериментів з метою оцінки запропонованих алгоритмів при використанні IT-інфраструктури діючого ЦОД дуже дорого. Для оцінки запропонованих алгоритмів методу рівномірної

консолідації ВМ з використанням ідеї імітації відпалу використовується інструментарій CloudSim [184]. Це модульний і розширюваний інструментарій з відкритим кодом, який має вбудовані можливості для реалізації та порівняння алгоритмів керування для різних хмарних середовищ і робочих навантажень. Розширена версія CloudSim використана при моделюванні [200] для врахування показників енергозбереження.

При моделюванні [194] створено ЦОД, що містить 800 гетерогенних ФС з двоядерними ЦП. Половина ФС є серверами HP ProLiant ML110 G4 з продуктивністю 1860 мільйонів інструкцій на секунду (MIPS), а інша половина складається з серверів HP ProLiant ML110 G5 з 2660 MIPS. Модель кожного ФС має пропускну здатність мережевого інтерфейсу 1 Гб/с. Характеристики ВМ відповідають типам екземплярів Amazon EC2, але з одним ядром через характер даних робочого навантаження. Дані робочого навантаження взяті з реального ЦОД і використані як робоче навантаження на ВМ при моделюванні. Дані робочого навантаження отримані в рамках проекту CoMon, інфраструктури моніторингу для PlanetLab. Інтервал утилізації вимірювань становить 5 хвилин [15].

Об'єм оперативної пам'яті для ВМ визначений у такий спосіб: середній екземпляр з високими вимогами до ЦП (2500 MIPS, 0,85 ГБ ОП); дуже великий екземпляр (2000 MIPS, 3,75 ГБ); малий екземпляр (1000 MIPS, 1,7 ГБ); дуже малий екземпляр (500 MIPS, 613 МБ). Спочатку ВМ розподіляються відповідно до вимог ресурсів, визначених типами ВМ [189].

Методи, представлені в роботі [200], модифіковані для реалізації алгоритмів OIB: getNewVmPlacement, getNewVmPlacementFromUnderUtilizedHost і findHostForVm з пакету org.cloudbus.cloudsim.power.

Результати моделювання отримані після проведення десяти експериментів. Для порівняння ефективності запропонованого алгоритму з вбудованими в

CloudSim, метрики з [194] використовуються, як представлено у розділі 5, (5.2)-(5.5).

Кожен експеримент повторюється десять разів для кожного набору даних навантаження. Остаточні результати усереднюються за цими експериментами. Результати моделювання для алгоритмів THR, IRQ, MAD, LRR та LR отримані з [194]. Метрики роботи обох алгоритмів OIB (5.2)-(5.5) обчислені і представлені в табл. 8.1 для порівняння.

Табл. 8.1. Результати моделювання порівняно з алгоритмами в [194] та обох алгоритмів OIB (середні значення)

Назва алгоритму	ESV ($x10^{-3}$)	E (kWh)	SLAV ($x10^{-5}$)	SLATAH %	PDM %
THR-MMT-0.8	4.19	89.92	4.57	4.61	0.10
IQR-MMT-1.5	4.00	90.13	4.51	4.64	0.10
MAD-MMT-2.5	3.94	87.67	4.48	4.65	0.10
LRR-MMT-1.2	2.43	87.93	2.77	3.98	0.07
LR-MMT-1.2	1.98	88.17	2.33	3.63	0.06
OIB (простий)	1.81	88.24	2.05	3.41	0.06
OIB (з обмеженнями)	1.71	88.4	1.93	3.85	0.05

Мета експериментальної фази полягала в оцінці показників якості алгоритмів OIB при різних умовах навантаження та порівнянні їх з результатами, отриманими в [194]. Результати, наведені в табл. 8.1, вказують на те, що при використанні обох алгоритмів OIB для моделювання управління ЦОД споживає майже таку ж кількість енергії, як і алгоритм LR-MMT-1.2, тому що алгоритми OIB використовують LR і MMT політики, але виконують тільки розміщення ВМ відповідно до техніки оптимізації відпалу. При використанні простого алгоритму OIB, метрика *PDM* не змінюється порівняно з алгоритмом LR-MMT-1.2, оскільки більш рівномірне завантаження ФС не впливає безпосередньо на кількість міграцій ВМ. Унаслідок використання обох алгоритмів OIB слід зазначити, що більш рівномірне завантаження ФС дозволяє зменшити метрику *SLATAH*. Використовуючи алгоритм OIB з обмеженнями міграцій, можна зменшити кількість міграцій і, як наслідок, зменшити метрику *PDM*. Рівномірне завантаження ФС призводить до зменшення порушень SLA через резервування

деяких ресурсів ФС для реагування на зростаючі випадкові вимоги до ресурсів у найближчому майбутньому.

8.3. Оцінка результатів моделювання двостадійного методу управління ресурсами хмарного центру оброблення даних на основі алгоритму променевого пошуку

Дослідження роботи двостадійного методу виконане у працях [280, 285, 323] на фрагментах набору даних GCT [106]. Реалізація двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку у вигляді діаграм класів застосунку моделювання наведена у додатку А.

Дані для тестування і дослідження методу обрані з наборів даних GCT і являють собою частину записів за певний діапазон часу. Це необхідно, щоб врахувати всі дані за один і той самий період часу з різних журналів GCT. З набору випадковим чином відібрано 70000 завдань з певними вимогами до ресурсів *k*. Для кожного завдання буде розгорнута окрема ВМ у процесі моделювання. Обираємо 6000 ФС, з яких на 5832 серверах розміщено ВМ, та на 168 серверах завдань немає, але вони доступні для розміщення ВМ. Моделювання виконане на комп’ютері з процесором i5-3570 та об’ємом системної пам’яті 4024 Мб;

Якісними показниками роботи алгоритму є кількість вимкнених ФС PM_{sleep} та кількість міграцій ВМ VM_{mig}, за допомогою яких ФС вивільнюються від ВМ. Крім того, враховано час роботи методу, *t*. Запропонований метод на основі моделі реалізований за допомогою мови C# [285]. Симуляції проводились на комп'ютері з процесором Intel i7-3632QM та 8 ГБ оперативної пам'яті під керуванням Windows 10 Pro 64bit. Інтерфейс програми імітації показаний на рис. 8.1. Перша серія експериментів проведена без обмеження на кількість одночасних міграцій в перерахунку на один ФС.

У табл. 8.2 представлені результати роботи запропонованого методу з консолідації ВМ для різних значень β. У табл. 8.3 представлені результати роботи методу при консолідації ВМ з використанням методики нижньої границі.

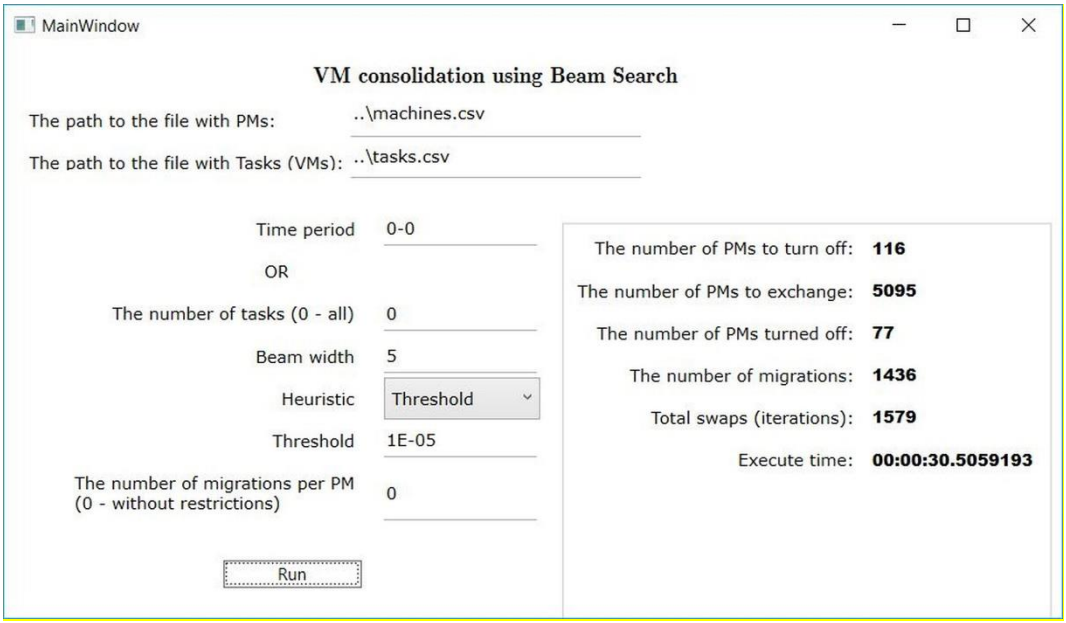


Рис. 8.1. Інтерфейс програми моделювання двостадійнго методу на основі променевого пошуку [285]

Кількість ФС, переключених в режим сну, позначений PM_{sleep}. Кількість міграцій ВМ позначена VM_{mig}. Час пошуку прийнятного рішення за допомогою двостадійного метода позначений T. Кількість ФС, переключених у сплячий режим і кількість міграцій ВМ вибирані як якісні показники запропонованого методу.

Табл. 8.2. Результати моделювання консолідації ВМ з використанням методики порогу вільних ресурсів

β	B	A	PM_{sleep}	VM_{mig}	T, s
0,005	199	3	0	0	0,034
0,004	299	3	0	0	0,052
0,003	1098	9	0	0	0,279
0,002	1393	10	0	0	0,38
0,001	1868	28	13	206	1,421
0,0009	1943	31	15	238	1,817
0,0008	2002	39	22	352	2,465
0,0007	2081	43	25	403	3,067
0,0006	2204	48	29	470	3,321
0,0005	2302	55	32	502	4,446
0,0004	2407	63	37	566	5,884
0,0003	2565	73	43	714	6,784
0,0002	2764	82	49	836	8,68
0,0001	3706	95	57	956	12,8
0,00005	4323	106	66	1128	18,422
0,00001	5095	116	77	1442	29,995
0	5328	125	82	1447	33,15

Табл. 8.3. Результати моделювання консолідації ВМ з використанням методики нижньої границі

B	A	PM_{sleep}	VM_{mig}	T, s
5707	125	81	1435	32,13

Вплив значення порогу β на якісні показники роботи алгоритму є суттєвим (рис. 8.2 та рис. 8.3), залежність виявилася майже лінійною. Чим менший поріг β тим більше ФС потрапляють на вхід алгоритму.

При $\beta = 0$ деякі ФС (а саме ті, у яких хоча б один ресурс повністю зайнятий) не потрапляють на вхід алгоритму, тому що при перевірці наявності ресурсів використовується строга нерівність. При $\beta < 0.0002$ алгоритм з методикою порогу вільних ресурсів починає працювати більш ефективно і на граничних значеннях працює краще чим з методикою нижньої границі.

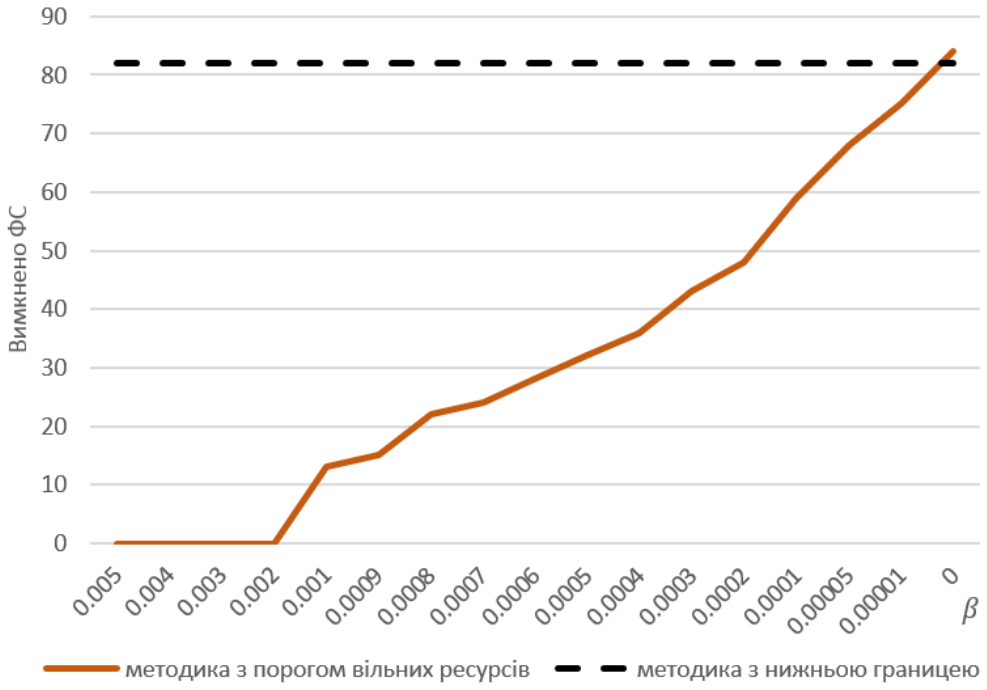


Рис. 8.2. Вплив порогу на кількість ФС, переключених в режим сну

Після формування списку A ефективність пошуку кінцевого стану кластера можливо оцінити за допомогою відношення кількості ФС, що заплановані для переключення у режим сну, до кількості ФС, що фактично визначені для перемикання в режим сну після роботи другої стадії методу. Ефективність перемикання ФС в режим сну зростає з ростом значення порогу β і знаходиться в діапазоні від 45% до 65%.

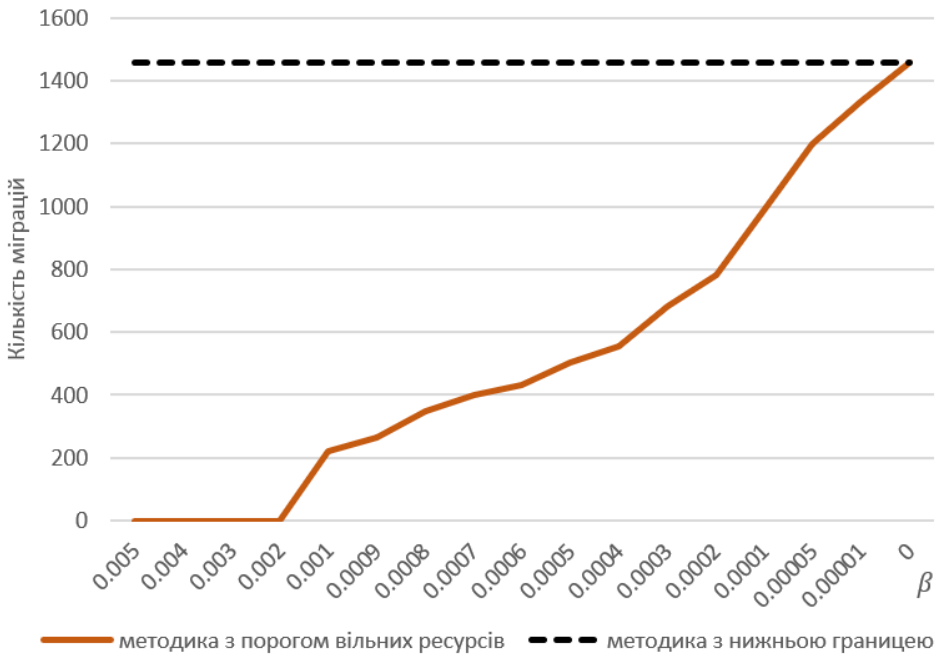


Рис. 8.3. Вплив порогу на кількість міграцій

Для перевірки роботи алгоритму з різними методиками та їх порівняння створено декілька наборів даних з однаковою кількістю ФС та завдань за інші періоди часу в наборі даних GCT. Унаслідок моделювання отримані показники ефективності, які розрізняються не більше ніж на 7%.

Перевагою використання методики з порогом вільних ресурсів є можливість адаптуватися до стану кластера через зміни порогу перед циклом управління, враховуючи інші ресурси, час міграції ВМ та інтенсивність надходження заявок на створення нових ВМ.

У процесі дослідження роботи методу, з метою встановлення впливу ширини променя на якісні показники роботи алгоритму, ширина променю для алгоритму променевого пошуку змінювалась в діапазоні від 1 до 15 (табл. 8.4 та табл. 8.5).

Табл. 8.4. Результати моделювання консолідації ВМ з використанням методики нижньої границі

n	PM_{sleep}	VM_{mig}	t, с
1	23	423	66.72
2	23	435	101.33
3	23	433	137.75
4	23	450	173.57
5	82	1460	508.09
6	82	1457	812.05

7	83	1471	832.64
8	81	1418	894.11
9	83	1479	1169.78
10	80	1417	-
11	82	1428	-
12	82	1449	-
13	85	1520	-
14	83	1501	-
15	84	1487	-

Табл. 8.5. Результати моделювання консолідації ВМ з використанням методики з порогом вільних ресурсів

n	PM_{sleep}	VM_{mig}	***t*, с**
1	27	422	53.51
2	35	548	116.5
3	57	986	186.26
4	57	959	204.49
5	57	940	247.78
6	57	918	331.03
7	55	922	383.36
8	59	969	466.55
9	57	943	492.38
10	56	957	471.13
11	56	944	549.1
12	58	972	584.52
13	60	1063	718.3
14	58	1021	791.87
15	61	1042	844.12

При використанні методики нижньої границі збільшення ширини променя алгоритму призводить до суттєвого збільшення часу пошуку рішення. Так, при ширині променя n=10 час виконання алгоритму значно зростає і стає неприпустимим для використання в кластері з високою динамікою процесів створення ВМ (табл. 8.4).

Крім того, зміна ширини променя при використанні кожної методики може не призводити до суттєвого покращення якісних показників, при суттєвому збільшенні часу пошуку. Так, для методики з нижньою границею зміна ширини променя з 5 до 15 не призводить до покращення якісних показників, а для

методики з порогом вільних ресурсів покращення якісних показників не відбувається при ширині променя від 3 до 15 (рис. 8.4 та рис. 8.5).

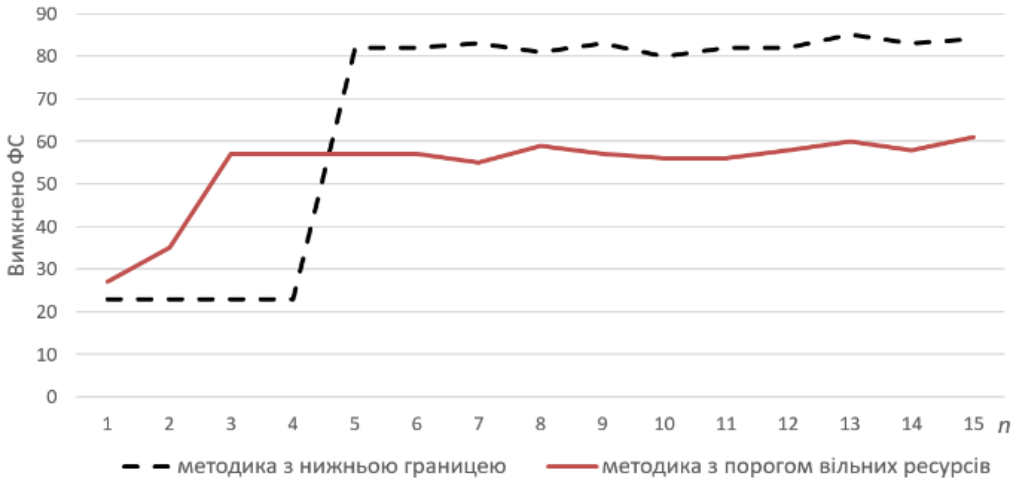


Рис. 8.4. Вплив ширини променя на кількість ФС, переключених в режим сну

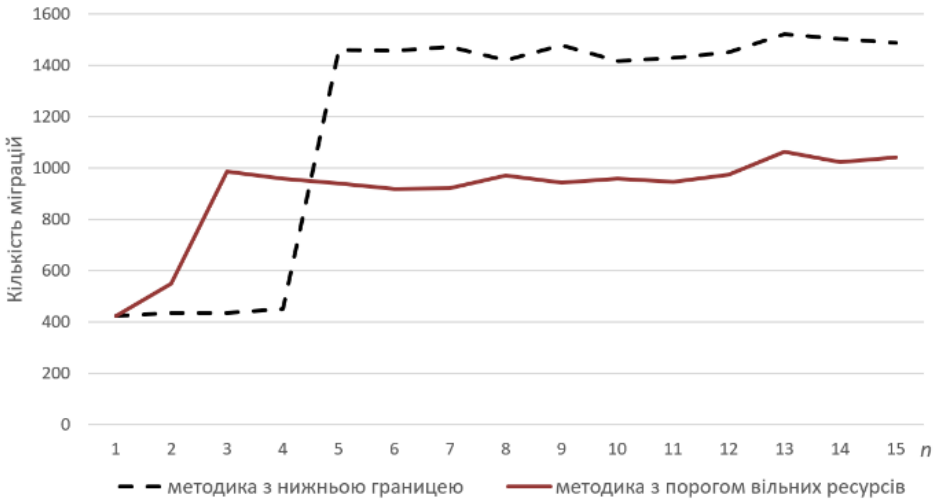


Рис. 8.5. Вплив ширини променя на кількість міграцій

Таким чином, треба звертати увагу на кількість міграцій, забезпечуючи їх допустиму кількість. З урахуванням висновків, рекомендується обирати ширину променя від 5 до 8, залежно від умов роботи методу [280].

Прийнятне рішення задачі консолідації ВМ з допомогою двостадійного методу променевого пошуку можливе при застосуванні обмежень на кількість міграцій, а також на час оптимізації. Результати показують, що обчислення за методикою нижньої границі може тривати близько 32 секунд. Така продуктивність може бути неприйнятною у реальних ЦОД з динамічним управлінням ВМ, оскільки система управління повинна враховувати динаміку нових процесів розгортання ВМ і одночасно міграцію ВМ.

Щоб зменшити кількість міграцій ВМ, рекомендується обмежити максимальну кількість одночасних міграцій з/на ФС. У реальних умовах роботи ЦОД гіпервізори обмежують максимальну кількість одночасних міграцій з/на ФС, на мережевий ресурс і на сховище даних, щоб забезпечити майже 100-відсоткову гарантію відсутності порушень SLA під час міграції ВМ [196]. На деяких робочих навантаженнях, які є інтенсивними або на старих пристроях, велика кількість одночасних міграцій (більше рекомендованого числа) може вичерпати системний ресурс ФС і призвести до простою ВМ [277].

Друга серія експериментів проведена з обмеженням на максимальну кількість одночасних міграцій з/на ФС. Це обмеження починає впливати на результати моделювання, коли межа одночасних міграцій з/на ФС дорівнює 2. У виробництві гіпервізори зазвичай налаштовані на значення в діапазоні 4-16, залежно від апаратних засобів [196]. Результати показують, що вплив цього обмеження з'являється, коли кількість ФС в наборі B є занадто малою, щоб приймати ФС від ФС з набору A. У таких випадках показник PMsleep зменшується приблизно на 40%.

Шість наборів даних з різним числом ФС та ВМ були створені на основі даних, отриманих з моніторингу використання кластера Google. ФС і завдання були взяті з різних часових діапазонів цих наборів для всебічної перевірки роботи алгоритму. Отримані результати показують ті ж показники продуктивності, які були отримані при першій конфігурації середовища моделювання.

8.4. Експериментальне дослідження методу динамічної консолідації і розміщення віртуальних машин на основі алгоритму навчання з підкріпленням

У дисертаційній роботі використовуються тільки три типи ресурсів (CPU, RAM, BW) для виконання моделювання з використанням НП [324, 330]. Такий вибір зумовлений використанням розширеної версії CloudSim [200] для забезпечення моделювання з урахуванням енергозбереження. Крім того, в якості вхідного набору даних для моделювання роботи ВМ використовуються дані

моніторингу реальних навантажень від Bitbrains [22]. Реалізація методу динамічної консолідації і розміщення віртуальних машин на основі алгоритму навчання з підкріпленням у вигляді діаграм класів застосунку моделювання наведена у додатку А.

Для оцінки запропонованого методу динамічної консолідації і розміщення ВМ на основі алгоритму навчання з підкріпленням проведено моделювання з використанням наборів даних моніторингу Bitbrains [24] і проаналізовано результати щодо значень метрик порушення SLA, споживання енергії та кількості міграцій ВМ під час моделювання. Також використовується середовище моделювання CloudSim [184] з необхідними модифікаціями. CloudSim є модульним і розширюваним інструментарієм з відкритим кодом, який має вбудовані можливості для реалізації та порівняння енергоощадних алгоритмів керування для різних хмарних середовищ і робочих навантажень. Моделювання проводилося на комп'ютері з процесором Intel i7-3632QM і 8 ГБ оперативної пам'яті під управлінням Windows 10 Pro 64bit.

В цій серії експериментів модель ЦОД містить 200 гетерогенних ФС з характеристиками, що показаними в табл. 8.6. Крім того, при моделюванні використано 500 гетерогенних ВМ чотирьох типів з різним числом ядер, як показано в табл. 8.7. Типи ВМ відповідають запитуваним ресурсам, які використовуються ВМ, що описані в [24]. Протягом експерименту кількість ВМ не змінювалася, але в загальному випадку кількість ВМ може змінюватися. Більш того, в динамічному середовищі деякі ВМ, визначені для міграції, можуть припинити існування під час кроку управління.

Табл. 8.6. Конфігурації ФС, використані при моделюванні

Тип ФС	Кіль-кість	MIPS of CPU	Кількість PE	RAM	Bandwidth, Gbit/s
Dell Inc. PowerEdge R640	50	2500	56	196608	1
Dell Inc. PowerEdge R740	50	2500	56	196608	1
Dell Inc. PowerEdge R830	50	2200	88	262144	1
Dell Inc. PowerEdge R940	50	2500	112	393216	1

Дані робочого навантаження з реального ЦОД, що реалізує приватну хмару [22], використовувалися як робоче навантаження на ВМ при моделюванні. Початкова версія набору даних Bitbrains містить 1200 трейсів ВМ в яких

збережено використання ресурсів різними серверами (Web/application/database). Загалом ВМ належать до 44 різних класів, причому кожен клас містить від 8 до 50 ВМ. Але лише 500 трейсів були використані як вхідні дані в дисертаційному дослідженні. У кожному файлі трейсу зберігаються дані моніторингу однієї ВМ про використання процесора, пам'яті та мережевого інтерфейсу. Усі експерименти проведені під динамічним навантаженням з 500 ВМ для періоду моделювання 9 днів, в якому інтервал вимірювань становить 300 секунд.

Табл. 8.7. Конфігурації ВМ, використані при моделюванні

PM Type	Number	MIPS of CPU	Number of PEs	RAM capacity	Bandwidth, Mbit/s
Type 1	125	500	2	2048	100
Type 2	125	1300	4	4096	100
Type 3	125	2200	8	8192	100
Type 4	125	2500	16	16384	100

При моделюванні, також модифіковані класи CloudSim таким чином, що під час моделювання враховуються три типи ресурсів (*CPU*, *RAM*, *BW*). Крім того, класи CloudSim були модифіковані (як показано в пунктах 5.9, 5.10) з урахуванням різних профілів енергоспоживання ФС при різних навантаженнях серверного процесора, наданих SPEC [232].

Метою фази моделювання є оцінка якісних показників роботи методу НП під різним навантаженням та порівняння їх з якісними показниками методів, запропонованих у [194].

Оцінка ефективності запропонованого методу НП виконана з урахуванням трьох показників якості: часу порушення SLA, кількості споживаної енергії та кількості міграцій ВМ. На першому етапі виявлено вплив швидкості навчання α і коефіцієнта дисконтування γ на ефективність запропонованого методу. На другому етапі досліджено вплив вагових коефіцієнтів β та δ на показники ефективності запропонованого методу. На третьому етапі порівнюється запропонований метод з методами і евристиками консолідації ВМ, представленими в [194]. Швидкість навчання змінювалась від 0,1 до 1 з кроком 0,25 і коефіцієнт дисконтування змінювався від 0,1 до 1 з кроком 0,25. Вагові коефіцієнти β і δ фіксувалися на рівні 0,5. Унаслідок 25 експериментів,

значення α і γ, що забезпечують мінімальний час порушення SLA були визначені у такий спосіб: $\alpha = 0.5$, $\gamma = 0.5$.

На другому етапі вагові коефіцієнти змінювалися від 0,1 до 1 з кроком 0,25 із дотриманням обмеження $\beta + \delta = 1$. Значення α і γ зберігалися фіксованими на рівні 0,5, як визначено на першому етапі. Унаслідок 11 експериментів значення вагових коефіцієнтів β і δ, що забезпечують мінімальний час порушення SLA були визначені у такий спосіб: $\beta = 0.8$, $\delta = 0.2$. Значення вагових коефіцієнтів β і δ, що забезпечують мінімальне споживання електроенергії, визначені у такий спосіб: $\beta = 1$, $\delta = 0$.

На третьому етапі проведено моделювання запропонованого методу НП та методів, запропонованих в [194], використовуючи раніше визначені значення α, γ, β і δ.

На рис. 8.6 показані сукупні значення показника порушення SLA, індикатора споживання енергії та кількості міграцій ВМ.

На кожному кроці управління поточне значення показника якості додається до попередніх, таким чином отримано накопичення по кожному показнику якості для всіх ФС. Результати, наведені на рис. 8.6, вказують на те, що запропонований метод перевершує конкурентні методи з [194] за показниками часу порушення SLA і кількості міграцій ВМ, оскільки запропонований метод виконує менше міграцій ВМ і переключає в режим сну менше ФС, ніж у конкурентних методах.

У результаті запропонований метод забезпечує менший час порушень SLA у процесі управління. Методи, представлені в [194], забезпечують низьке енергоспоживання за рахунок набагато більшої кількості міграцій, що зазвичай неприйнятно в умовах реального ЦОД [196]. Більш того, конкурентні методи не враховують енергоспоживання при перемиканні ФС в режим сну і навпаки. Це може бути серйозним недоліком, коли велика кількість ФС часто змінює свої стани [230].

Аналіз результатів. Результати моделювання свідчать про хороші показники якості роботи конкурентних методів [194] при статичній конфігурації

і фіксованим числом ФС і ВМ. Але кількість ВМ у реальному ЦОД може змінюватися з часом. Таким чином, існують два недоліки політик, запропонованих в [194], які виконують просту мінімізацію споживання енергії: (i) висока затримка при плануванні створення нової ВМ через відсутність доступних ФС, що задовольняють вимогам з боку ВМ, (ii) збільшення споживання енергії при перемикання ФС з активного стану в сплячий режим і навпаки (режим налаштування ФС [228]) та коли кількість ВМ часто змінюється протягом відносно короткого періоду часу.

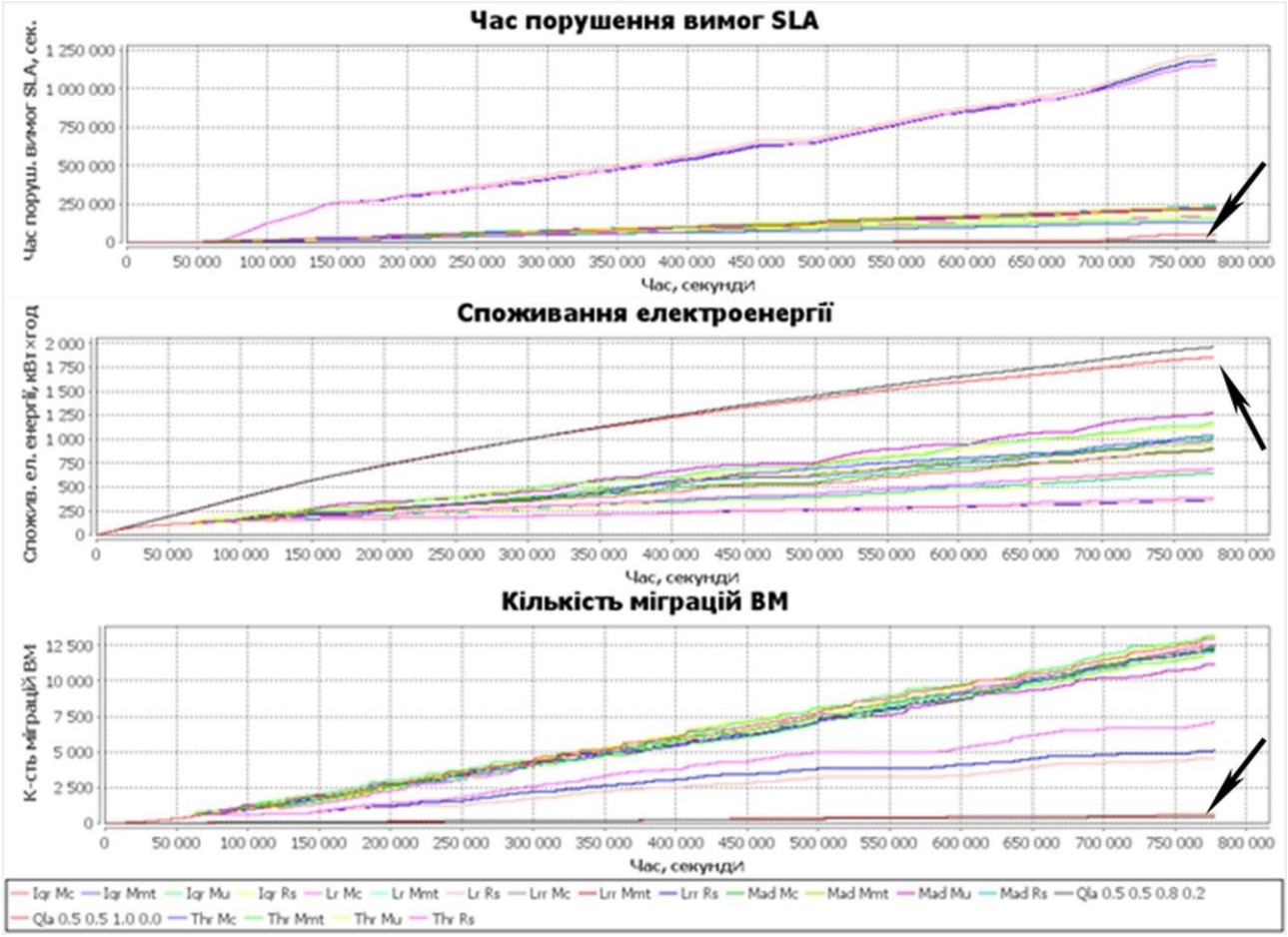


Рис. 8.6. Сукупні значення показників якості для запропонованого методу (Qla) і конкурентних методів [194]

З іншого боку, політика постійно включених ФС (англ. AlwaysOn) [233] забезпечує мінімальну затримку планування нової ВМ, але призводить до максимального споживання енергії. Політика AlwaysOn – це статична політика керування живленням, яка широко використовується постачальниками хмарних послуг. Ця політика постійно підтримує фіксовану кількість ФС в активному стані. При цьому, важливою проблемою є пошук балансу між політикою

AlwaysOn та політикою, запропонованою в [194]. Але вирішення цієї проблеми є предметом майбутніх досліджень.

Режим налаштування ФС вимагає значного часу, коли ФС переходить з сплячого режиму в активний режим. Час, який витрачається на перемикання ФС між сплячим режимом і активним режимом, називається часом налаштування. Чим вище час налаштування, тим вище затримка планування нової ВМ, коли немає активних ФС, які можуть задовольнити запит планування нової ВМ, або міграції ВМ з перевантажених ФС. Час налаштування для деяких виробничих серверів [234] представлено в табл. 8.8.

Час налаштування ФС змінюється від 20 секунд до 200 секунд залежно від апаратної та програмної конфігурації і може досягати 260 секунд [228]. Споживання енергії під час режиму налаштування близька до максимальної норми для ФС [229].

Табл. 8.8. Час налаштування для деяких конфігурацій ФС [234]

Модель ФС	Режим налаштування				
	OFF - IDLE	SLEEP - IDLE	IDLE - SLEEP	SLEEP - OFF	IDLE - OFF
HpProLiantMl110G3PentiumD930	50	20	3	2	5
HpProLiantMl110G4Xeon3040	45	10	3	2	5
HpProLiantMl110G5Xeon3075	50	20	3	2	5
IbmX3250XeonX3470	45	10	3	2	4
IbmX3250XeonX3480	45	10	2	2	4
IbmX3550XeonX5670	60	20	3	2	5
IbmX3550XeonX5675	75	30	3	2	5

Аналогічні результати повідомлялися в [226, 227]. За час режиму налаштування ФС витрачають близько 200 Вт [227]. Таким чином, велика тривалість режиму налаштування ФС змушує провайдерів хмарних ЦОД застосовувати будь-які динамічні політики керування живленням для перемикання конкретних серверів у сплячий режим при падінні робочого навантаження.

Іншим важливим висновком представлених результатів моделювання є вплив кількості вхідних параметрів, які враховуються під час моделювання в середовищі CloudSim. Конкурентні методи, описані в [194], враховують тільки робоче навантаження на ЦП [15] як вхідний сигнал для кожної модельованої ВМ.

У дисертаційному дослідженні використовуються три типи ресурсів (*CPU*, *RAM*, *BW*) як вхід кожної модельованої ВМ. Таким чином, отримані в [194] результати відрізняються від результатів, отриманих модифікованим інструментарієм CloudSim, з точки зору споживання енергії та часу порушення SLA.

Таким чином, основними перевагами запропонованого методу навчання з підкріпленням є: можливість скоротити час порушення SLA, що може бути дуже дорогим у виробництві; можливість обслуговувати більше запитів на створення нових ВМ, особливо коли кількість ВМ часто змінюється; можливість зменшити навантаження на мережу ЦОД через зменшення кількості міграцій.

8.5. Експериментальне дослідження адаптивного методу комбінованого прогнозування

Оцінка початкових даних. Для прогнозування та аналізу роботи адаптивного методу прогнозування, який представлено у працях [286, 327], використаний набір даних, отриманий системою моніторингу ВМ розподіленого хмарного ЦОД Bitbrains [22]. Набір даних Bitbrains містить показники роботи 1750 ВМ із розподіленого ЦОД, який спеціалізується на постачанні послуг хостингу та бізнес-розрахунків для підприємств. Кожен файл набору даних (трейс) містить часовий ряд у вигляді показників споживання однією ВМ основних обчислювальних ресурсів, таких як процесорний час, обсяг оперативної пам’яті, продуктивність підсистеми зберігання та продуктивність підсистеми мережевої взаємодії [24]. Для дослідження роботи запропонованого методу з кожного трейсу взяті тільки дані про споживання процесорних ресурсів. Дані про споживання процесорного ресурсу ВМ уявляють собою часовий ряд. Тривалість ряду складає один місяць, заміри значень зроблені з інтервалом 5 хвилин. Однак, при виконанні обчислень для перевірки запропонованих методів використано дані за чотири доби, тобто 1152 значення часового ряду. Реалізація методу усередненого комбінованого прогнозування і методу зваженого комбінованого прогнозування наведена у додатку Б.

Враховуючи, що навантаження на кожну ВМ мають свій, особливий характер, з усього набору даних для виконання досліджень обрано 6 часових

рядів з навантаженням на центральний процесор для ВМ з номерами 599, 795, 840, 850, 904 та 957.

Часові ряди з навантаженням на ЦП ВМ обрані таким чином, щоб статистичні показники кожного з них суттєво відрізнялись. Статистичні показники кожного трейсу наведені в табл. 8.9. Трейси не включають ніяку інформацію про навантаження і застосунки, що виконуються всередині ВМ.

Табл. 8.9. Статистичні показники часових рядів споживання процесорного ресурсу

Покажчик	ВМ599	ВМ795	ВМ840	ВМ850	ВМ904	ВМ957
Середнє	10.37598	0.83502	25.63766	5.44349	1.20278	1.75686
Стандартна похибка	0.82311	0.03790	0.02682	0.08682	0.01092	0.07531
Медіана	1	0.5	25.63333	4.63333	1	1.11667
Мода	1	0.5	25.83333	4.3	1	1
Стандартне відхилення	27.84026	1.28200	0.90721	2.93650	0.36940	2.54714
Дисперсія	775.08024	1.64354	0.82303	8.62302	0.13645	6.48794
Ексцес	5.10384	41.56920	-0.07084	18.90430	15.00060	36.97383
Асиметричність	2.65870	5.78652	0.04550	3.94964	3.29004	5.37852
Інтервал	95.88333	14.20000	5.76667	26.06667	3.53333	31.01667
Мінімум	1	0.5	22.8	2.03333	1	1
Максимум	96.88333	14.7	28.56667	28.1	4.53333	32.01667
Сума	11870.11667	955.26557	29329.48004	6227.35615	1375.97619	2009.84549
Кількість вимірювань	1152	1152	1152	1152	1152	1152

За допомогою візуального аналізу часових рядів кожної ВМ (рис. 8.7), не використовуючи формальні методи перевірки на стаціонарність, можна зробити висновок, що наведені часові ряди є нестаціонарними. Але можна припустити, що на деяких відрізках часу в кожному з трейсів часовий ряд є стаціонарним, тому виправданим є використання методів, наведених у розділі 4.5. Крім того, дослідити всі можливі трейси і отримати відповідні моделі прогнозу не уявляється можливим, тому запропонований метод не потребує попереднього аналізу ряду на стаціонарність.

Підбір моделей для прогнозування споживання процесорного ресурсу та розроблення програм на основі запропонованих у розділі 4 методів [286, 327] виконана мовою програмування R [17] та пакету *forecast* [25]. Запуск програм та експерименти з часовими рядами проведені з використанням комп’ютера на базі

процесора Intel i7-3632QM, з об’ємом пам’яті 8 GB під управлінням Windows 10 Pro 64bit. Для запуску альтернативних методів використано мову програмування R версії 3.4.4 (2018-03-15) [29].

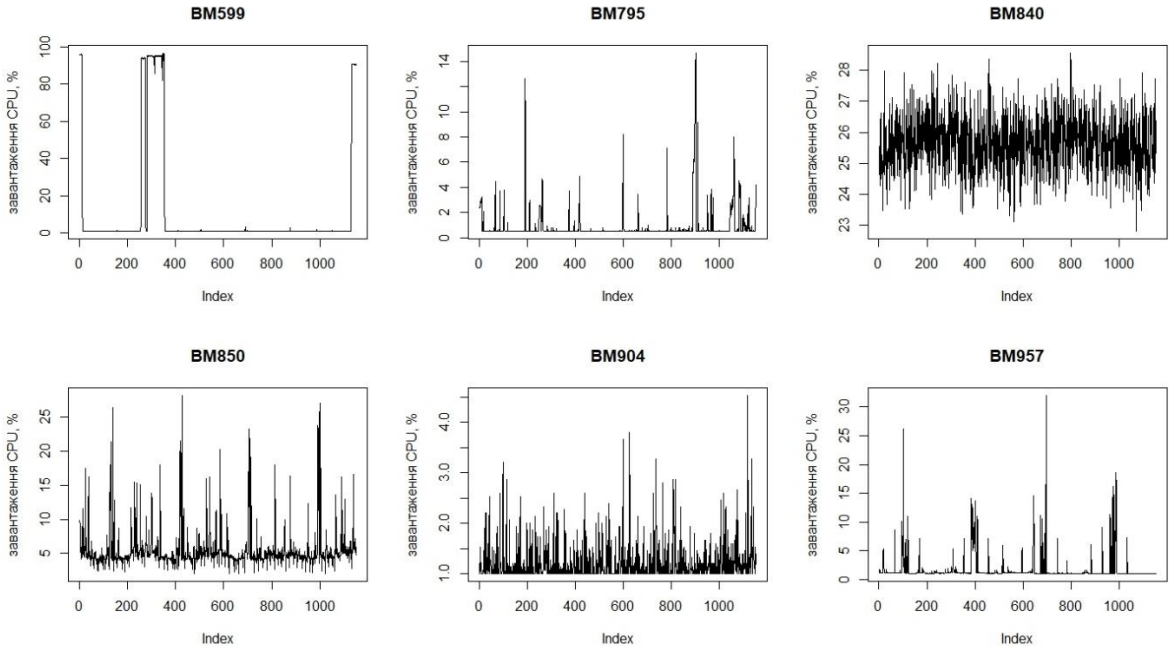


Рис. 8.7. Часові ряди вимірів навантаження на процесорний ресурс ВМ

Коефіцієнти моделі для альтернативних методів, описаних у розділі 4, оцінюються з використанням функцій пакету *forecast* [23, 25, 27] як показано у табл. 8.10.

Табл. 8.10. Пояснення функцій R, що використовуються в адаптивному методі прогнозування

Метод прогнозу-вання	R функція	Опис
SES	*fitses* = *ses*(*ts*, *h*=1) *F* = *fitses*$*mean*[1]	*fitses* – прогноз на один крок вперед, розрахований за допомогою простого експоненціального згладжування
ARIMA	*ARIMAfit* = *auto.arima*(*ts*,*lambda*=0, *biasadj*=TRUE) *F* = *forecast*(*ARIMAfit*, *h*=1, *level*=95)	*ARIMAfit* – модель ARIMA визначена за допомогою іункції *auto.arima*()

Метод прогнозу-вання	R функція	Опис
HOLT	*fitholt = holt*(*ts, h*=1) *F = fitholt*$*mean*[1]	*fitholt* – прогноз на один крок вперед, розрахований за допомогою методу Хольта з лінійним трендом
DHOLT	*fitdholt = holt*(*ts, damped*=TRUE, *phi*=0.9, *h*=1) *F = fitdholt*$*mean*[1]	*fitdholt* – прогноз на один крок вперед, розрахований за допомогою методу Хольта с демпіруваним трендом
LR	*fitLRt = tslm*(*ts ~ trend*) *fcLRt = forecast*(*fitLRt, horizon*) *F = fcLRt*$*mean*[1]	*fitLRt* – є моделлю лінійної регресії з трендом
TBATS	*fittbats = tbats*(*ts, biasadj*=TRUE) *fcTBATS = forecast*(*fittbats, h=horizon*) *F = fcTBATS*[["*mean*"]]	*fittbats* – є моделлю TBATS, яка ідентифікована методом, запропонованим в [28]

Похибка прогнозу MAPE обчислюється з використанням функції *accuracy*(*F,O*) в R, де *F* позначає прогнозоване значення на один крок, а *O* означає спостережуване значення, отримане з підсистеми моніторингу. У табл. 8.10 *ts* позначає часовий ряд, *horizon* позначає кількість кроків прогнозу, *lambda*, *biasadj, h, level, damped, phi, trend* – це параметри функції *forecast*.

8.5.1. Спрощений адаптивний метод прогнозування

Для перевірки працездатності, запропонований адаптивний метод спрощений через виконання першого кроку вручну. Таким чином, роботу обраних альтернативних методів прогнозування, описаних у розділі 4.5, досліджено на прикладі шести часових рядів (рис. 8.7). Унаслідок аналізу показників якості обрано один метод прогнозу з альтернативних, який використовувався на другому кроці роботи спрощеного адаптивного методу прогнозування (САМП). Відповідні параметри моделі для цього методу підбирались через вибір розміру навчальної вибірки. При цьому, на другому

кроці роботи спрощеного методу інші методи прогнозування вже не розглядалися.

Аналіз альтернативних методів прогнозування $A_1 = \{\text{SES, ARIMA, HOLT, DHOLT}\}$ проведений для змінного розміру вікна тренувальних даних W_1={8..62}. Ідея полягає в тому, щоб на кожному кроці управління обчислювати параметри моделей прогнозу для декількох розмірів вікон тренувальних даних, як показано на рис. 8.8. При цьому розмір максимального вікна тренувальних даних $w_{\max}$ і розмір мінімального вікна тренувальних даних $w_{\min}$ визначається через підбір. Дослідження показали, що при розмірі вікна менше ніж 8 якість прогнозу обраними методами є незадовільною.

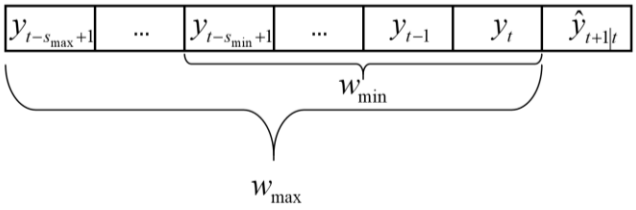


Рис. 8.8. Зміна довжини вікна тренувальних даних

Для кожного з обраних методів підібрані моделі прогнозу для 30 вікон тренувальних даних (від 8 до 62 з кроком 2), $W_1 = \{8..62\}$, і для шести трейсів роботи різних ВМ. Для кожної з 30 моделей обчислене значення MAPE і побудовано залежність помилки прогнозу MAPE від довжини вікна тренувальних даних. Результати досліджень спрощеного адаптивного методу представлені на рис. 8.9–рис. 8.14.

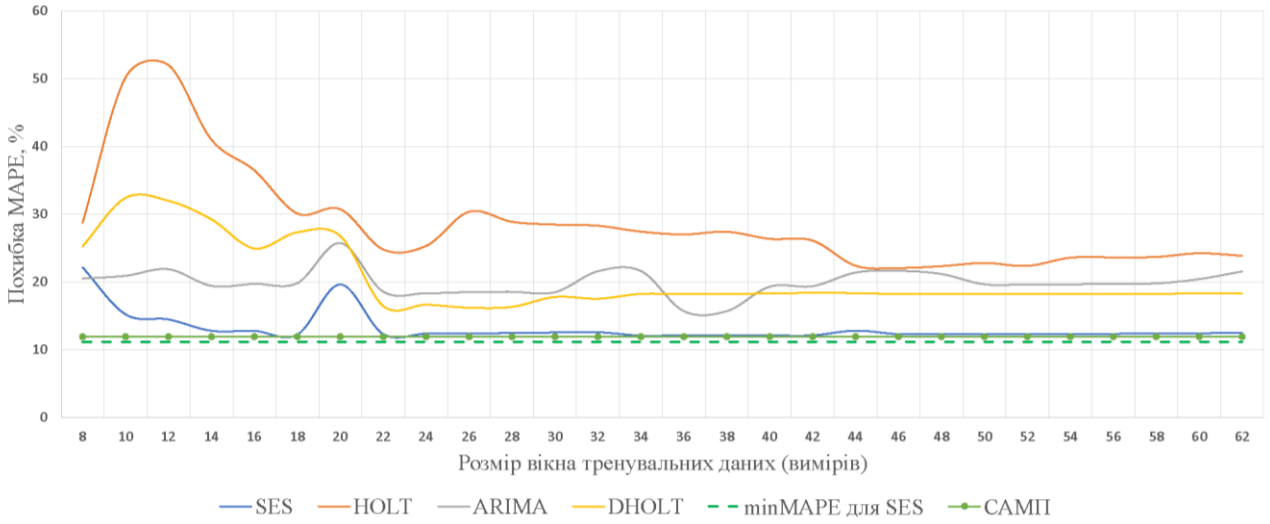


Рис. 8.9. Залежність точності прогнозу споживання процесорного ресурсу від довжини вікна тренувальних даних при використанні методів A_1 та САМП для часового ряду BM599

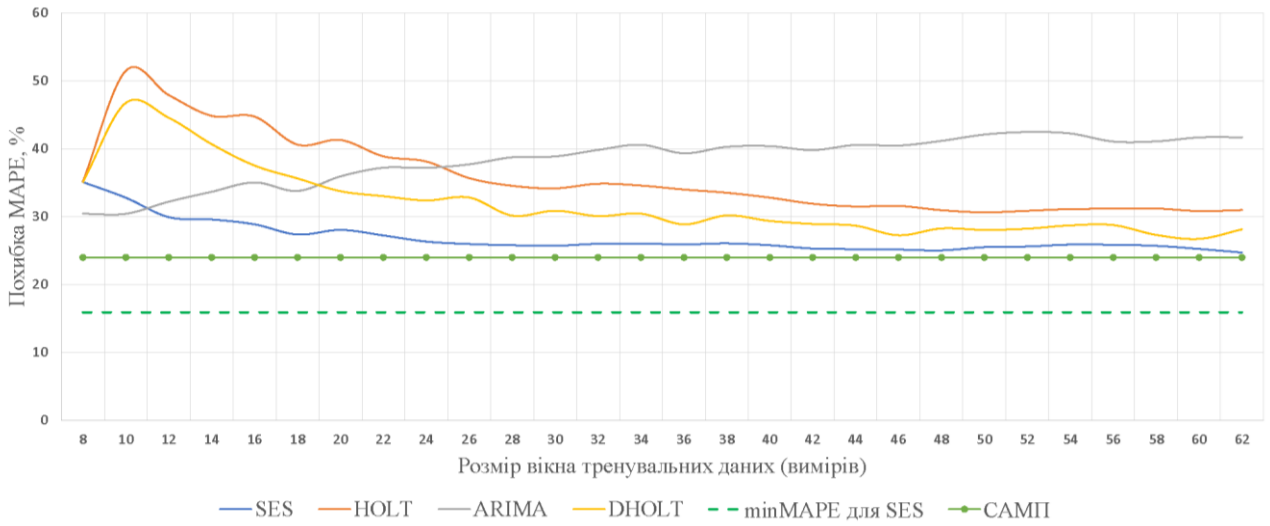


Рис. 8.10. Залежність точності прогнозу споживання процесорного ресурсу від довжини вікна тренувальних даних при використанні методів A_1 та САМП для часового ряду BM795

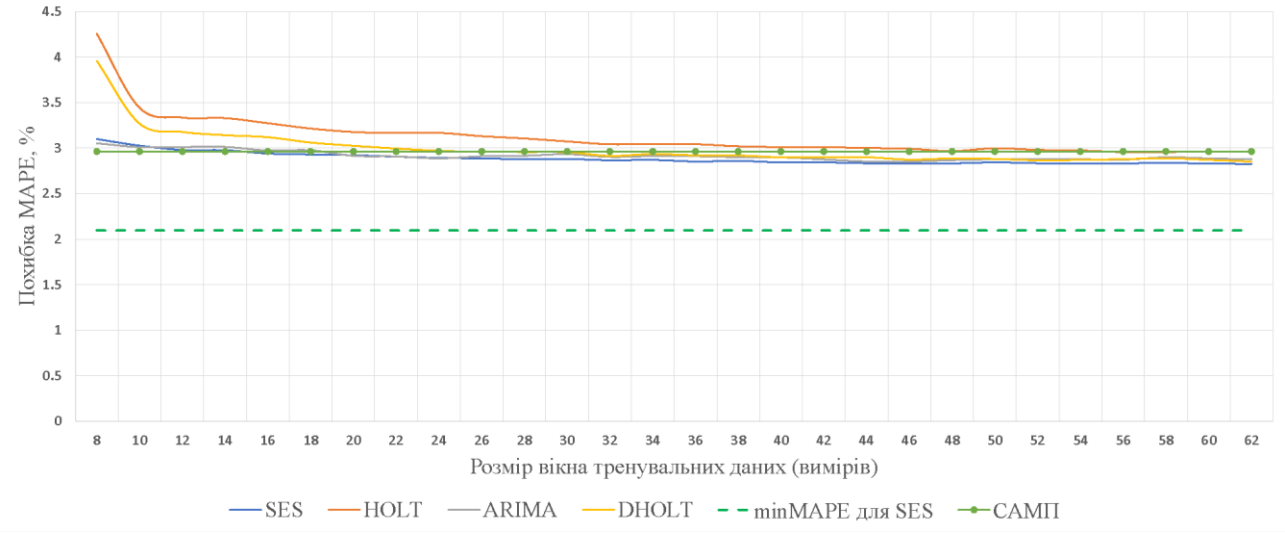


Рис. 8.11. Залежність точності прогнозу споживання процесорного ресурсу від довжини вікна тренувальних даних при використанні методів A_1 та САМП для часового ряду BM840

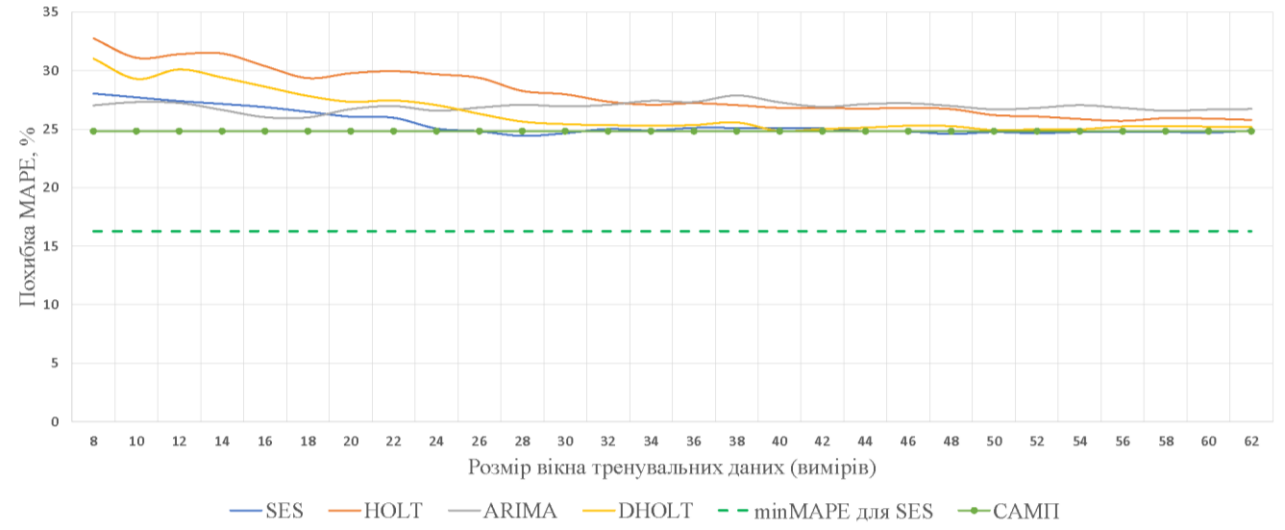


Рис. 8.12. Залежність точності прогнозу споживання процесорного ресурсу від довжини вікна тренувальних даних при використанні методів A_1 та САМП для часового ряду BM850

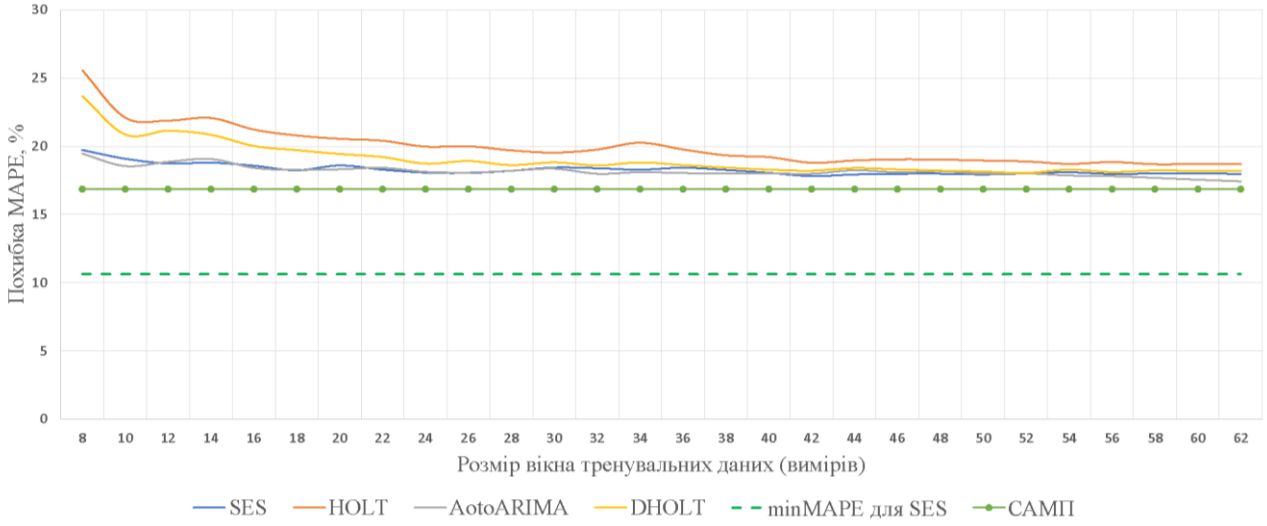


Рис. 8.13. Залежність точності прогнозу споживання процесорного ресурсу від довжини вікна тренувальних даних при використанні методів A_1 та САМП для часового ряду BM904

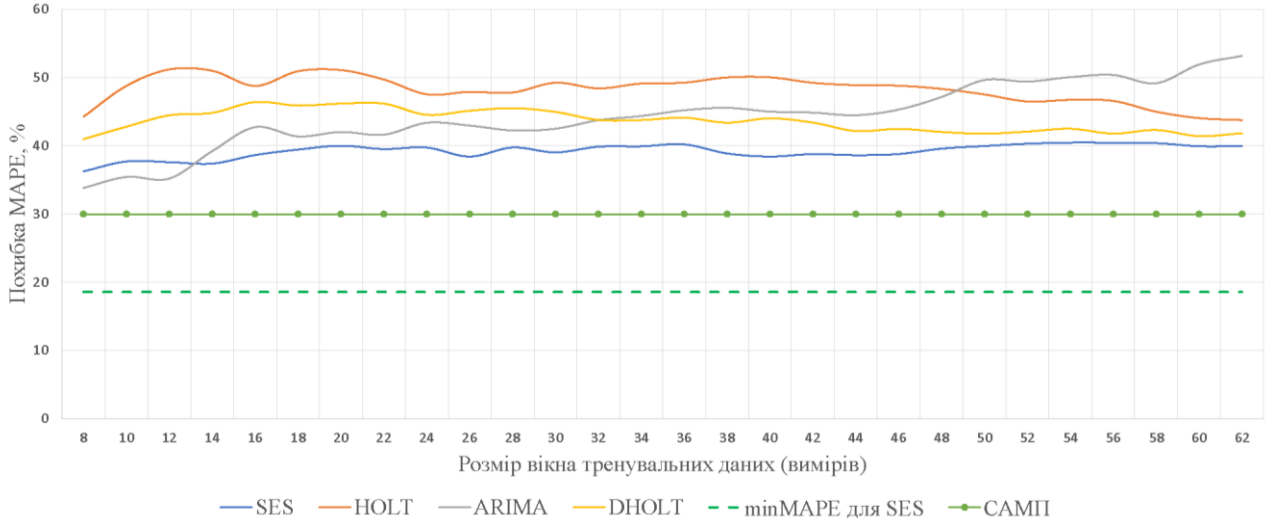


Рис. 8.14. Залежність точності прогнозу споживання процесорного ресурсу від довжини вікна тренувальних даних при використанні методів A_1 та САМП для часового ряду BM957

Результати аналізу показують, що якість прогнозу в значній мірі залежить від статистичних характеристик досліджуваного часового ряду. Цей висновок підтверджує припущення, що неможливо обрати один метод прогнозу для використання в промислових умовах роботи фізичних і віртуальних серверів. Крім того, з рисунків рис. 8.9 – рис. 8.14 можна бачити, що подальше збільшення розміру вікна тренувальних даних не призведе до суттєвого покращення якості прогнозу. Але, вірогідно, може впливати на результати прогнозування при використанні іншої множини альтернативних методів A_k.

Для обраних шести трейсів і для різних розмірів вікон тренувальних даних найбільш точні прогнози, серед досліджуваних методів, отримані за допомогою моделей методу SES. Для трейсу BM840 SES показав дуже гарний результат, а для трейсів BM599 та BM904 показав добрий результат. Однак для трейсів BM795, BM850 та BM957 прогноз виявився задовільним, що потребує додаткових заходів щодо покращення прогнозів. Таким чином, для перевірки роботи спрощеного адаптивного методу прогнозування споживання ресурсів на другому кроці його роботи використано метод простого експоненційного згладжування з адаптацією вікна тренувальних даних.

Для дослідження ефективності другого кроку роботи спрощеного методу обчислено мінімальну похибку (minMAPE) серед всіх 30-ти розмірів вікон тренувальних даних для кожного трейсу. Тобто, на кожному кроці управління для підбору моделі прогнозу обирається розмір вікна тренувальних даних, при якому похибка MAPE є мінімальною, потім обчислюється середня похибка для всіх кроків управління, від 63 до 1151. Значення minMAPE обчислено з метою порівняння зі значеннями, отриманими на другому кроці спрощеного адаптивного методу.

Далі, на кожному кроці управління обирається прогнозоване значення, отримане моделлю, параметри якої обчислені за допомогою розміру вікна тренувальних даних, який дав мінімальну похибку на попередньому кроці управління. Відповідно, обчислено середню похибку (на графіках позначена САМП) для всіх кроків управління, від 63 до 1151. Для часових рядів BM795, BM904 та BM957 спрощений адаптивний метод показує кращі результати порівняно зі звичайним методом SES (без підбору вікна), відповідно на 10.7%, 7.8% і 23.6% в середньому прогноз більш точний. А для часових рядів BM599, BM840 та BM850 при розмірі вікна тренувальних даних більше ніж 24 точність прогнозу наближена до точності прогнозу з використанням звичайного SES, відповідно на 9.3%, 2.7% та 2.4% в середньому прогноз більш точний.

Таким чином, спрощений адаптивний метод прогнозування споживання обчислювальних ресурсів, що базується на використанні тільки одного методу прогнозування з альтернативних (в даному випадку це метод SES), дозволяє

отримати прогнозовані значення з меншою помилкою прогнозу, чим при використанні одного методу прогнозу з фіксованим розміром тренувальних даних для обчислення моделі прогнозу.

Результати кількісного аналізу показують, що точність прогнозу, отриманого за допомогою запропонованого методу, зростає в середньому від 2.4% до 23.6% залежно від часового ряду. Найкращий прогноз споживання процесорного ресурсу отриманий за допомогою методу простого експоненційного згладжування.

Унаслідок аналізу першої групи експериментів, яка призначена для виявлення впливу розміру вікна тренувальних даних на точність однокрокового прогнозу, можливо зробити такі висновки.

Висновок 1. Для всіх часових рядів, крім BM957, спостерігається зворотна залежність похибки MAPE від розміру вікна тренувальних даних. Тобто, зі збільшенням вікна тренувальних даних від 4 до 30 похибка MAPE зменшується досить сильно. У діапазоні розмірів вікна тренувальних даних від 30 до 60 MAPE зменшується дуже слабо. Виключення спостерігається при опрацюванні часового ряду BM957, де кращі значення MAPE досягаються за рахунок використання невеликих розмірів вікна тренувальних даних. Отже, при обчисленні прогнозів запропонованим адаптивним методом треба використовувати весь діапазон допустимих розмірів вікна тренувальних даних.

Висновок 2. Якісні характеристики часових рядів теж впливають на точність прогнозу. Якщо значення завантаження ЦП або довго не змінюється (як в часовому ряді BM599), або змінюються в невеликому інтервалі (як в часових рядах BM840 та BM904), то точність прогнозу висока або добра. Якщо спостерігаються викиди високої амплітуди (тобто інтервал більше чим 10, (табл. 8.9), і ці викиди виникають досить часто (через 20-50 вимірів), то похибка MAPE зростає і потрапляє в діапазон задовільних значень.

Висновок 3. Перша серія експериментів зі спрощеним адаптивним методом дозволила визначити метод, який дозволяє отримати найбільш точні прогнози. Серед альтернативних методів SES, HOLT, DHOLT та AutoARIMA найбільш точні прогнози (в середньому для кожного досліджуваного часового ряду)

можливо отримати методом SES з вікном тренувальних даних від 30 до 60. Але на певних частках досліджуваних часових рядів більш точний прогноз можливо отримати і іншими методами.

8.5.2. Адаптивний метод комбінованого прогнозування

Наразі, виникає два підходи для подальшого покращення точності прогнозу. Перший підхід полягає в обранні певної системи $< A_k, W_k >$, які найчастіше давали прогнози високої точності, і використовувати їх залежно від характеристик часового ряду на певних часових ділянках. Другий підхід полягає в адаптуванні до поточних умов роботи ЦП (або споживання іншого ресурсу) через вибір методу a_i і вікна тренувальних даних w_j (одної або декількох їх комбінацій $\{a_i, w_j\}$), які на попередньому кроці управління дозволили отримати прогноз з найбільшою точністю. Перший підхід може надати можливість отримувати прогнози з високою і доброю точністю тільки для часових рядів, які близькі за характеристиками до досліджуваних часових рядів. Для часових рядів з відмінними характеристиками, очевидно, точність прогнозу передбачити неможливо. Таким чином, перший підхід не виправдовує себе.

Для дослідження другого підходу проведена друга група експериментів з тими самими часовими рядами, але з більшою кількістю альтернативних методів. Друга група експериментів проведена з використанням іншого складу альтернативних методів прогнозування $A_2 = \{\text{SES, ARIMA, HOLT, DHOLT, LR, TBATS}\}$ проведений для змінного розміру вікна тренувальних даних $W_2 = \{4..80\}$. Метою проведення другої групи експериментів є визначення точності прогнозу запропонованого адаптивного методу прогнозування, а також визначити точність прогнозу при комбінуванні прогнозів, обчислених декількома методами.

Для кожного з альтернативних методів A_2 підібрані моделі прогнозу для 77 вікон тренувальних даних (від 4 до 80 з кроком 1) і для шести часових рядів з показниками завантаження процесора різних ВМ. На кожному кроці управління оцінені моделі прогнозу для розмірів вікон тренувальних даних, як показано на

рис. 8.8. Таким чином, мінімальний розмір досліджуваного вікна тренувальних даних в цій групі експериментів становить $w_{\min} = 4$, максимальний розмір вікна тренувальних даних становить $w_{\max} = 80$.

Ідея комбінування прогнозів сформульована у пункті 4.6 і полягає у такому. На поточному кроці управління від підсистеми моніторингу надходять актуальні виміри завантаженості ЦП. Обчислюються похибки MAPE прогнозів, що були обчислені на попередньому кроці, і формується список конфігурацій $K_{t|t-1} = \{a_i, w_j, Q_{i,j}\}$. Отриманий список сортується у порядку збільшення значення $Q_{i,j}$. Далі, на поточному кроці для системи управління обчислюється прогноз $\hat{y}^1_{t+1|t}$ із застосуванням методу і вікна з конфігурації $C^1_{t|t-1} = \{a^1_i, w^1_j, Q^1_{i,j}\}$ (4.2) як потенційний найкращий прогноз. Конфігурація з $p = 1$ є найкращою для використання на поточному кроці. Також, обчислюються прогнози з використанням методів *a* і вікон *w* з конфігурацій $C^p_{t|t-1}$, де $p = \overline{2,k}$, з метою обчислення усередненого комбінованого прогнозу $\hat{y}^c_{t+1|t}$ (4.3) і зваженого комбінованого прогнозу (4.4) згідно методики п. 4.6. Вибір параметру *k* обґрунтовано далі за результатами досліджень. Кількість попередніх кроків для обчислення зваженого комбінованого прогнозу обрана *m*=5.

Для дослідження впливу *k* на точність прогнозу обчислено комбіновані прогнози для кожного $k = \{2..5\}$. Результати обчислень похибки MAPE показані на рис. 8.15. У порівнянні з першою серією експериментів (рис. 8.9–рис. 8.14) точність прогнозу, отриманого адаптивним методом, дещо нижча, чим точність прогнозу, отриманого спрощеним адаптивним методом. Це обумовлено тим, що конфігурація $C^1_{t|t-1}$, що дала найточніший прогноз на попередньому кроці, дуже слабо відрізняється значенням *Q* від наступної конфігурації $C^2_{t|t-1}$ у відсортованому списку. Але значення *a* і *w* в кожній конфігурації відсортованого списку можуть значно відрізнятися, що сильно впливає на точність прогнозу на наступному кроці. Таким чином, вибираючи для обчислення комбінованого

прогнозу перші найкращі k пар $\{a_i^p, w_j^p\}$, $p=\overline{1,k}$, стає можливим покращити точність прогнозу.

На рис. 8.15 використані такі умовні позначення. "First Best Method" позначає адаптивний метод прогнозування, що використовує тільки одну пару $\{a_i^1, w_j^1\}$ з найкращої конфігурації $C_{t|t-1}^1$ у відсортованому списку. "MAPE First Best Method" – це похибка прогнозу (середня) методом і вікном, які дали найкращий прогноз на попередньому кроці. "MAPE Combined 2" позначає похибку комбінованого прогнозу з використанням *двох* прогнозів, k=2, отриманих з допомогою методів і вікон з конфігурацій $C_{t|t-1}^1$ і $C_{t|t-1}^2$, $C_{t|t-1}^2 \left| Q_{t|t-1}^1 < Q_{t|t-1}^2 \right.$. "MAPE Combined 3", "MAPE Combined 4" і "MAPE Combined 5" позначають похибки комбінованого прогнозу з використанням k=3, k=4, k=5. "MAPE Weighted" позначає помилку зваженого комбінованого методу прогнозування.

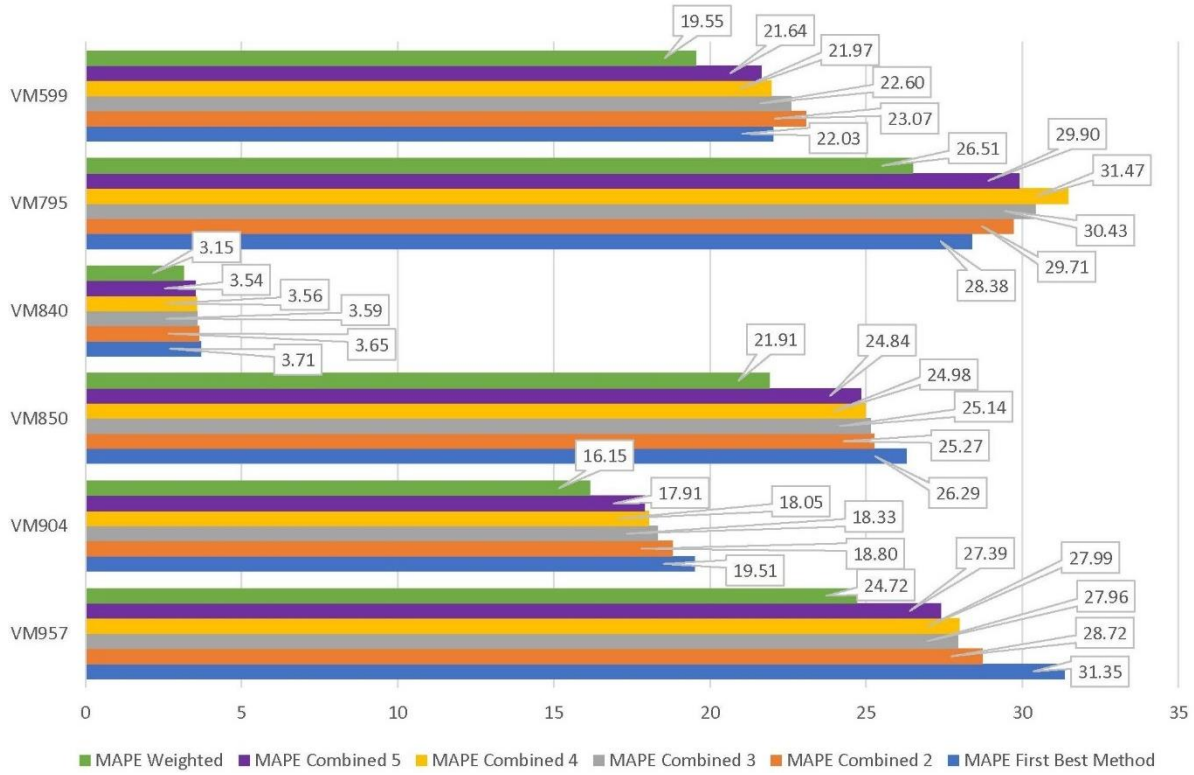


Рис. 8.15. Середня похибка MAPE для часових рядів BM599 – BM957

Нагадаємо, що похибки прогнозів попереднього кроку сортуються за збільшенням похибки MAPE і на поточному кроці використовуються перша пара

$\{a_i^1, w_j^1\}$ з найменшою похибкою (адаптивний метод прогнозу) і ще чотири пари з початку списку використовуються для обчислення комбінованого прогнозу, таким чином реалізуючи комбінований адаптивний метод прогнозу. На рис. 8.15 спостерігається тенденція, що чим більше *k*, тим більш точний прогноз можливо отримати у порівнянні з адаптивним методом, який базується тільки на одному прогнозі.

Для часового ряду BM957 точність прогнозу підвищилась на 12,6%, для часового ряду BM904 – на 8,2%, для часового ряду BM850 – на 5,5%, для часового ряду BM840 – на 4,7%, для часового ряду BM599 – на 1,8%. Однак для часового ряду BM795 точність прогнозу знизилась на 5,4%. Це пояснюється великим значенням ексцесу та відносно великим значенням асиметричності часового ряду. Таку ж саму тенденцію можливо бачити на рис. 8.16, що ілюструє стандартне відхилення середньої похибки MAPE для часових рядів BM599 – BM957.

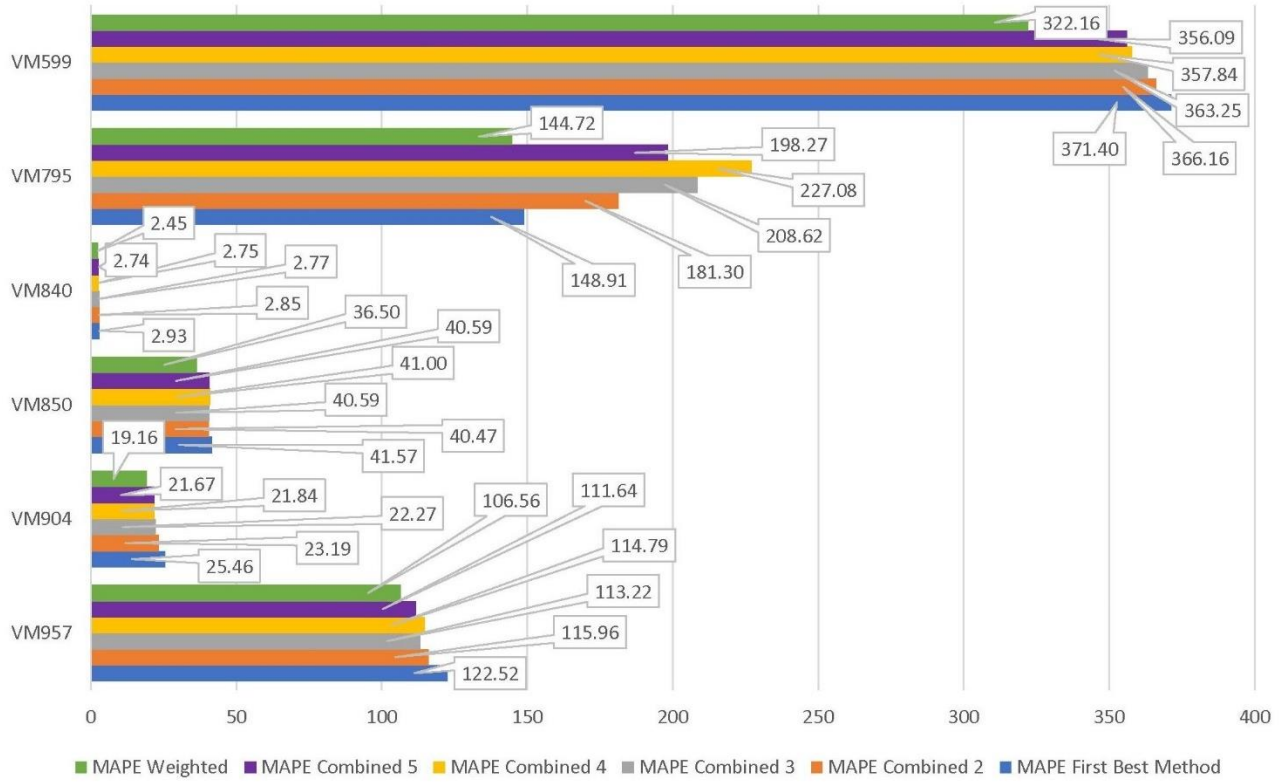


Рис. 8.16. Стандартне відхилення середньої похибка MAPE для часових рядів BM599 – BM957

Метод зваженого комбінованого прогнозування дає більш точні прогнози порівняно з адаптивним методом, який ґрунтується тільки на одному прогнозі.

Для часового ряду VM957 точність прогнозу збільшилася на 21,2%, для часового ряду VM904 – на 17,2%, для часового ряду VM850 – на 16,7%, для часового ряду VM840 – на 15,1%, для часового ряду VM795 – на 6,6%, для часового ряду VM599 – на 11,3%. Таким чином, метод зваженого комбінованого прогнозування також перевершує усереднений комбінований метод прогнозування.

Ще одною метрикою, яка дозволяє порівняти точність прогнозу спрощеного адаптивного методу прогнозування з його повною реалізацією є кількість незадовільних прогнозів, тобто прогнозів, при яких MAPE>50% (табл. 8.11, рис. 8.17). Для часового ряду VM795 кількість незадовільних прогнозів зменшилася на 5,5%, для часового ряду VM850 – зменшилася на 10,7%, для часового ряду VM957 – зменшилася на 20,7%, для часового ряду VM840 – не змінилася, для часового ряду VM599 – зменшилися на 17,6%, а для часового ряду VM904 – зменшилися на 26,7%. З цього можливо зробити висновок, що комбінація альтернативних методів прогнозування може призвести до поодиноких випадків, коли якісні показники прогнозу можуть погіршитися у порівнянні з якимось одним методом прогнозування.

Табл. 8.11 Якісні показники прогнозів, отриманих спрощеним адаптивним методом на базі SES, адаптивним методом прогнозування (усередненим комбінованим методом прогнозування і зваженим комбінованим методом прогнозування)

		Віртуальні машини (часові ряди)					
		599	795	840	850	904	957
спрощений адаптивний метод	Середня похибка MAPE	11.927	23.966	2.962	24.833	16.848	29.958
	Стандартне відхилення середньої похибки	265.093	109.394	2.287	40.51	16.612	100.675
	Кількість незадовільних прогнозів (похибка MAPE>50%)	15	104	0	156	41	138
адаптивний метод	Середня похибка MAPE	22.029	28.378	3.709	26.291	19.515	31.355
	Стандартне відхилення середньої похибки	371.401	148.908	2.929	41.573	25.455	122.517
	Кількість незадовільних прогнозів (похибка MAPE>50%)	17	91	0	150	86	116

адаптивний усереднений комбінований метод	Середня похибка MAPE	21.642	29.902	3.535	24.839	17.91	27.389
	Стандартне відхилення середньої похибки	356.092	198.273	2.736	40.593	21.674	111.645
	Кількість незадовільних прогнозів (похибка MAPE>50%)	16	103	0	141	73	105
адаптивний зважений комбінований метод	Середня похибка MAPE	19.546	26.507	3.15	21.91	16.152	24.716
	Стандартне відхилення середньої похибки	322.157	144.721	2.453	36.504	19.156	106.563
	Кількість незадовільних прогнозів (похибка MAPE>50%)	14	86	0	134	63	92

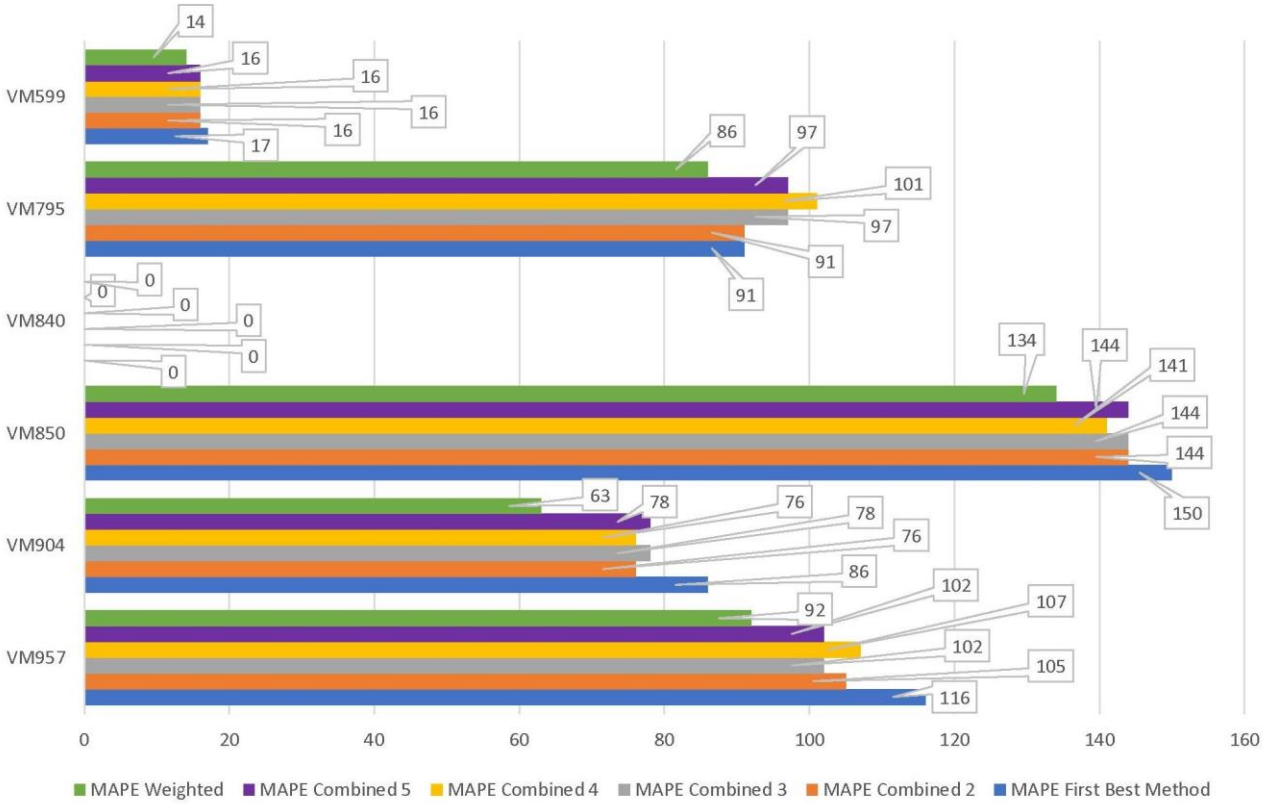


Рис. 8.17. Кількість незадовільних похибок (MAPE>50%) при прогнозуванні на всьому часовому ряді (для рядів BM599 – BM957)

Важливим результатом другої групи експериментів є, також, кількість обрання певного методу і певного вікна тренувальних даних, які дозволили отримати найточніший прогноз на кожному кроці управління, рис. 8.18 і рис. 8.19.

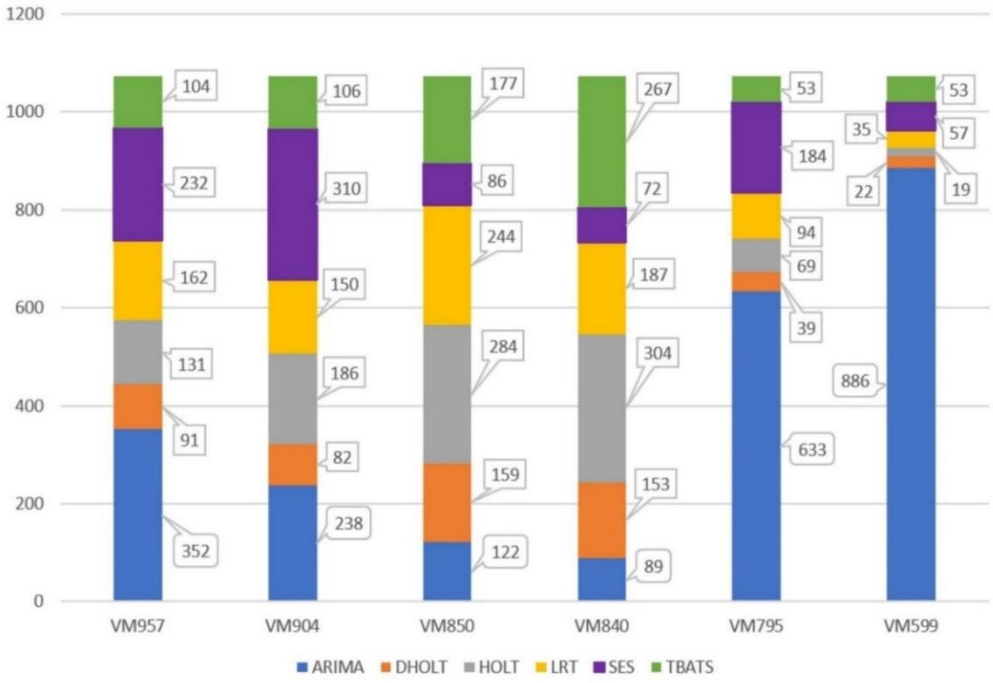


Рис. 8.18. Методи, що давали найкращий прогноз для кожного часового ряду (незалежно від розміру вікна тренувальних даних)

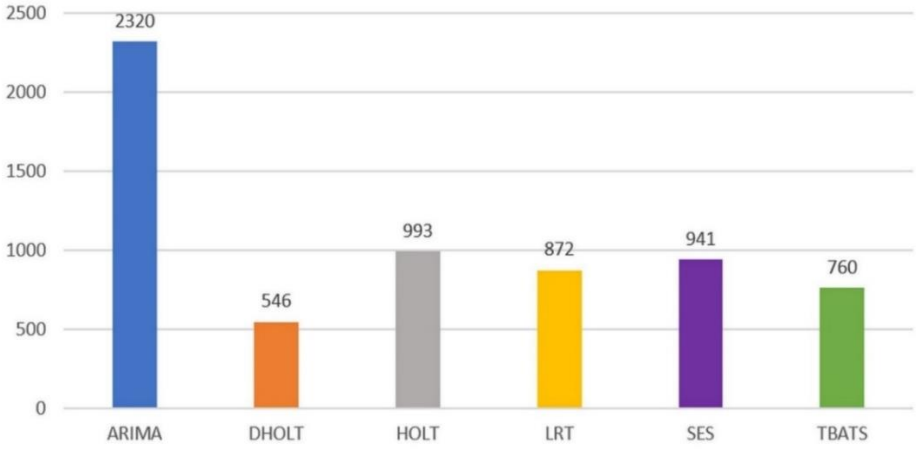


Рис. 8.19. Кількість методів, за допомогою яких отримані найточніші прогнози для всіх часових рядів (без урахування розміру вікна тренувальних даних)

Найбільш часто кращу точність прогнозу продемонстрував метод ARIMA. Але така велика кількість точних прогнозів не пояснюється його ефективністю. Такий стан відносно інших досліджуваних методів пояснюється тим, що часові ряди BM599 і BM795 складаються з великої кількості однакових значень, розташованих послідовно. Тому, будь-яка комбінація $\{a_i, w_j\}$ починає давати точний прогноз. Але унаслідок сортування на перше місце у відсортованому списку найчастіше попадає конфігурація з методом ARIMA. Усі інші методи застосовувались для отримання прогнозів у приблизно рівній кількості.

Виключенням є метод DHOLT, який вираховував найкращий прогноз найменшу кількість разів. Таким чином, для заощадження ресурсів цей метод можливо виключити з множини *A* альтернативних методів прогнозування.

На рис. 8.18 показана кількість використання альтернативних методів прогнозування для кожного часового ряду при роботі адаптивного методу прогнозування. Частота використання альтернативних методів для всіх часових рядів показана на рис. 8.19. На рис. 8.20 показана кількість використання певних розмірів вікна тренувальних даних для всіх часових рядів.

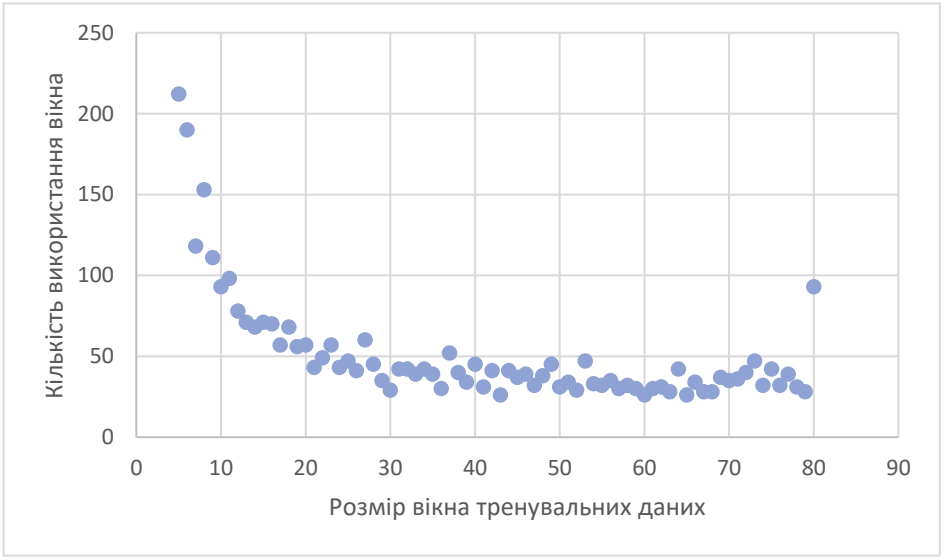


Рис. 8.20. Кількість разів використання певного розміру вікна тренувальних даних для всіх часових рядів (сумарно) при яких був отриманий найкращий прогноз

Унаслідок аналізу представлених даних зроблений висновок, що найчастіше точний прогноз був отриманий з використанням розміру вікна тренувальних даних від 12 до 80 значень. Ці показники можуть бути використані при покращенні адаптивного методу з метою прискорення часу його роботи та досягнення більшої точності.

Якісні показники роботи спрощеного адаптивного методу прогнозування, звичайного адаптивного методу прогнозування та адаптивного методу комбінованого прогнозування представлені у табл. 8.11.

8.5.3. Оцінка продуктивності роботи альтернативних методів прогнозування

Оскільки в запропонованому адаптивному методі використовуються бібліотеки мови R для отримання коефіцієнтів моделей прогнозу, то необхідно оцінити час виконання методів і забезпечити отримання прогнозу до наступного кроку реалізації управляючих впливів. Аналіз літературних джерел та технічної документації провідних компаній в галузі віртуалізації показав, що дані моніторингу стану віртуалізованої інфраструктури надходять з інтервалом 5 або 10 с. Також, поширеним є надання даних моніторингу з інтервалом 5 хвилин. Крім того, при використанні сучасних мережевих технологій управляючі впливи для міграції ВМ можуть тривати від 5 до 10 с. Інші управляючі впливи (увімкнення ВМ або ФС) потребують ще більшого часу. Таким чином потрібно отримати прогноз використання ресурсу за час, менший чим 10 с.

Для оцінки часу виконання альтернативних методів розроблено алгоритм і програму мовою R, яка базується на використанні функції Sys.time() і вимірює час роботи кожного методу при обчисленні прогнозу на наступний крок. Результати 100 кроків прогнозування усереднені і представлені на рис. 8.21–рис. 8.26. У табл. 8.12 представлені усереднені дані після 100 запусків методів для обчислення прогнозів з вікном тренувальних даних від 4 до 80 включно.

Табл. 8.12. Сумарний час, витрачений кожним методом на обчислення прогнозів з використанням вікон тренувальних даних від 4 до 80

	Час виконання методу, с						
Часовий ряд	**ARIMA**	**SES**	**HOLT**	**DHOLT**	**LR**	**TBATS**	**Сума**
BM957	10.2005	0.24556	0.35681	0.42985	0.58489	60.9224	72.74
BM850	6.86328	0.23134	0.36806	0.43341	0.55836	56.6125	65.067
BM840	5.53466	0.2464	0.4058	0.4698	0.55896	50.1974	57.4131
BM904	6.1153	0.24811	0.39424	0.4646	0.58842	66.297	74.1077
BM599	5.32975	0.2368	0.34613	0.40664	0.56972	49.3677	56.2567
BM795	6.30474	0.23046	0.33536	0.40602	0.56598	50.2982	58.1408

Експеримент проведений для всіх досліджуваних часових рядів. Наприклад, щоб порахувати 77 прогнозів, використовуючи вікна тренувальних даних від 4 до 80 з кроком 1, на наступний крок для часового ряду BM850 метод

ARIMA витрачає в середньому 6,86 с (усереднені дані для 100 повторів). Таким чином, для часового ряду BM850 альтернативні методи витратили на обчислення прогнозів в середньому 65 с. Результати показують, що витрати часу на отримання прогнозу методом TBATS в умовах оперативного управління хмарним ЦОД виявились неприйнятними. Однак для довгострокового управління ємністю ЦОД, коли дані моніторингу надходять кожні 300 с, цей метод може бути використаний. Також, доволі повільно працює метод ARIMA. Виключати цей метод з множини альтернативних недоцільно, так як він часто дає самий точний прогноз. Загалом, всі альтернативні методи, крім TBATS, можуть витратити значний час на отримання прогнозу (від 6 с. до 12 с. у табл. 8.12). Щоб зменшити витрати часу і обчислити прогнози за час, менший за 10 с. пропонується обчислювати прогноз з розміром вікна від 4 до 80 з кроком 2.

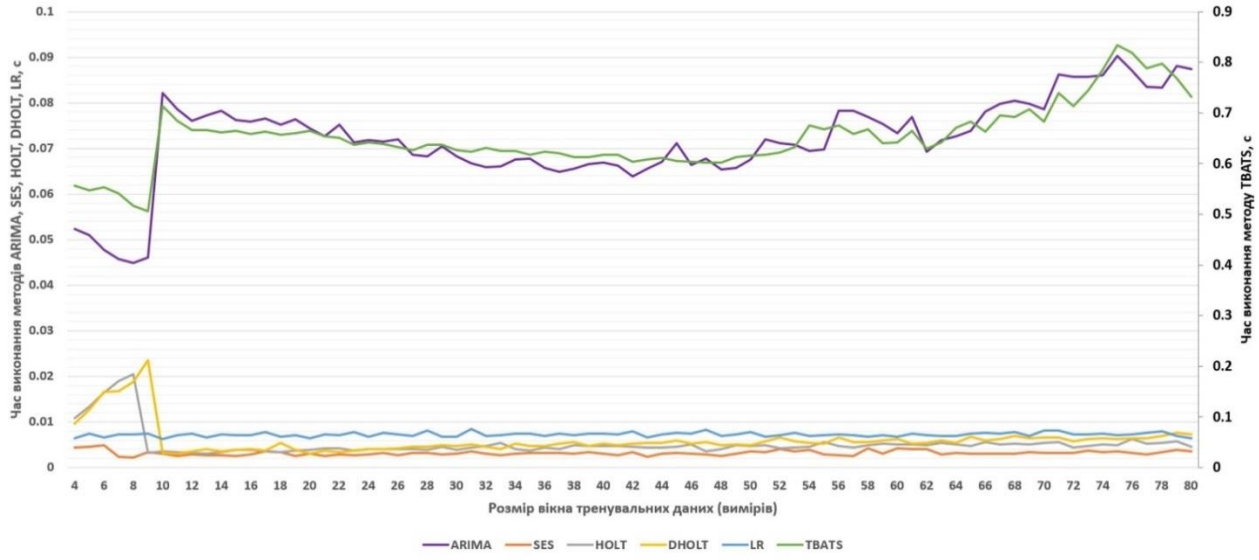


Рис. 8.21. Час роботи альтернативних методів при використанні різних розмірів вікна тренувальних даних для часового ряду BM840

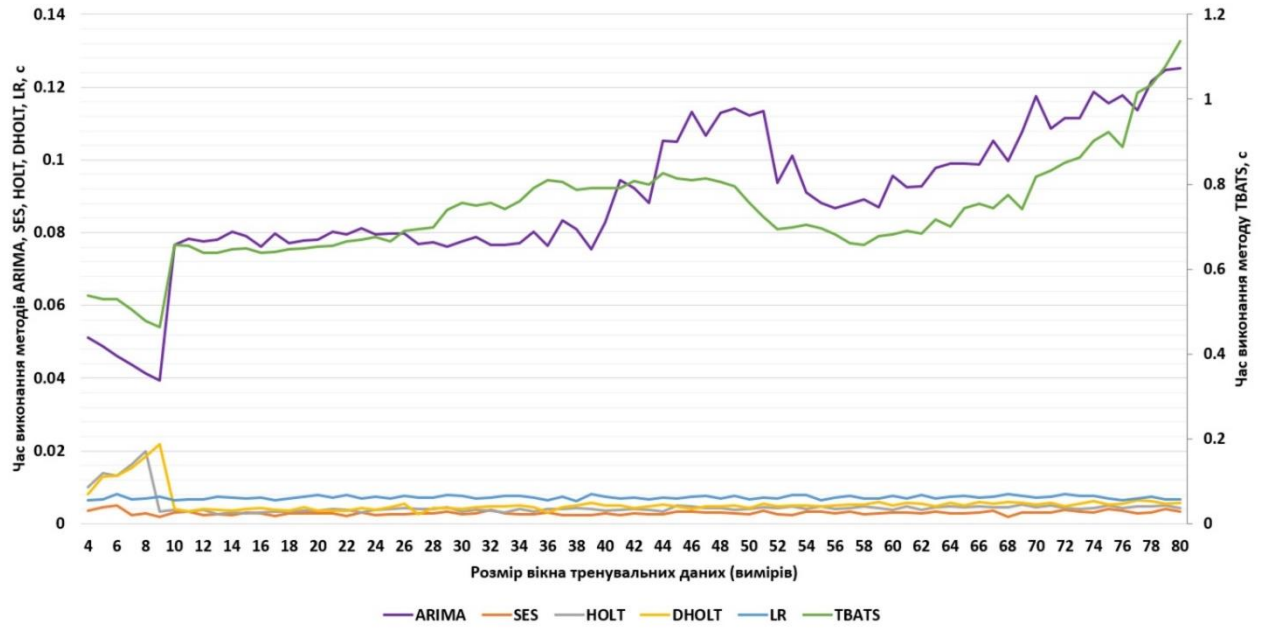


Рис. 8.22. Час роботи альтернативних методів при використанні різних розмірів вікна тренувальних даних для часового ряду BM850

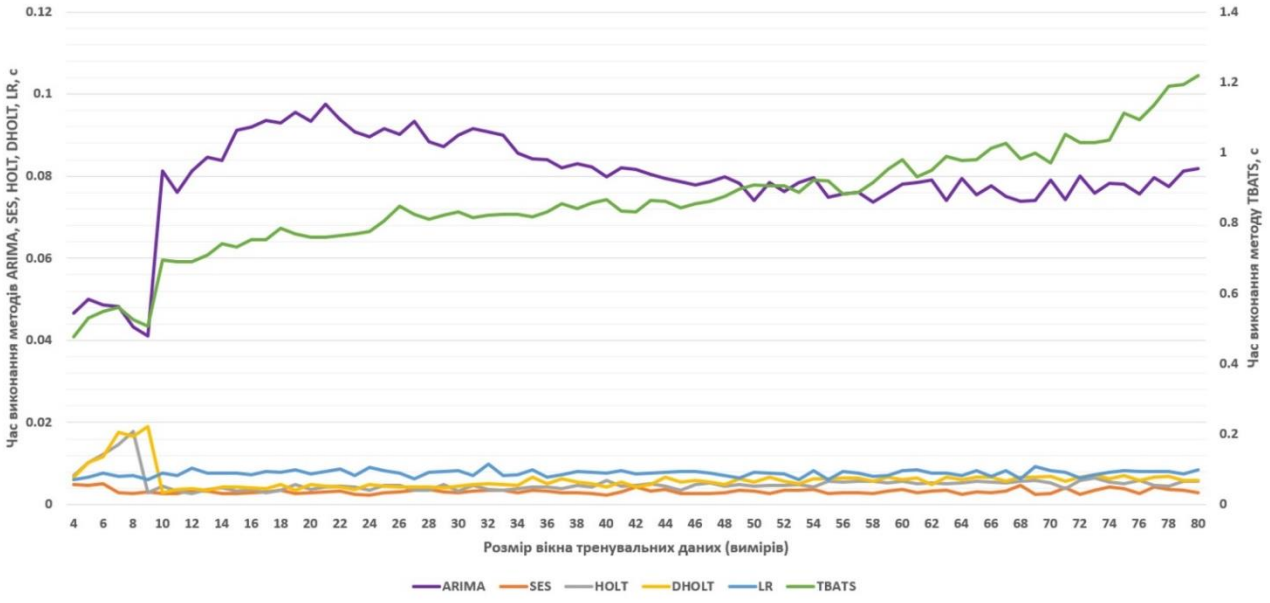


Рис. 8.23. Час роботи альтернативних методів при використанні різних розмірів вікна тренувальних даних для часового ряду BM904

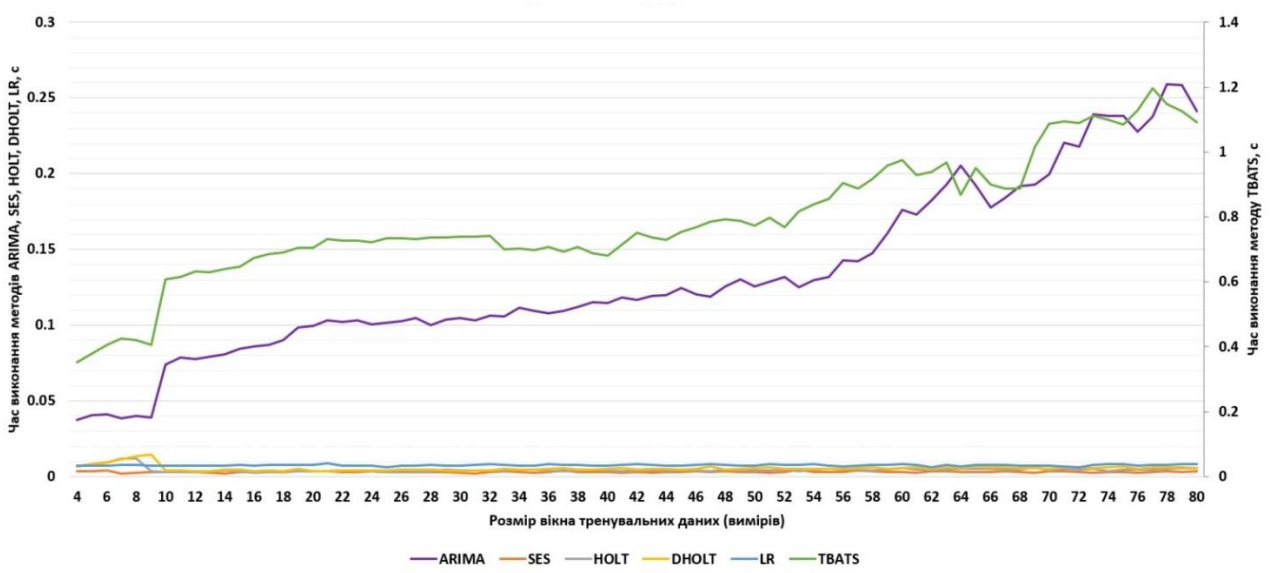


Рис. 8.24. Час роботи альтернативних методів при використанні різних розмірів вікна тренувальних даних для часового ряду BM957

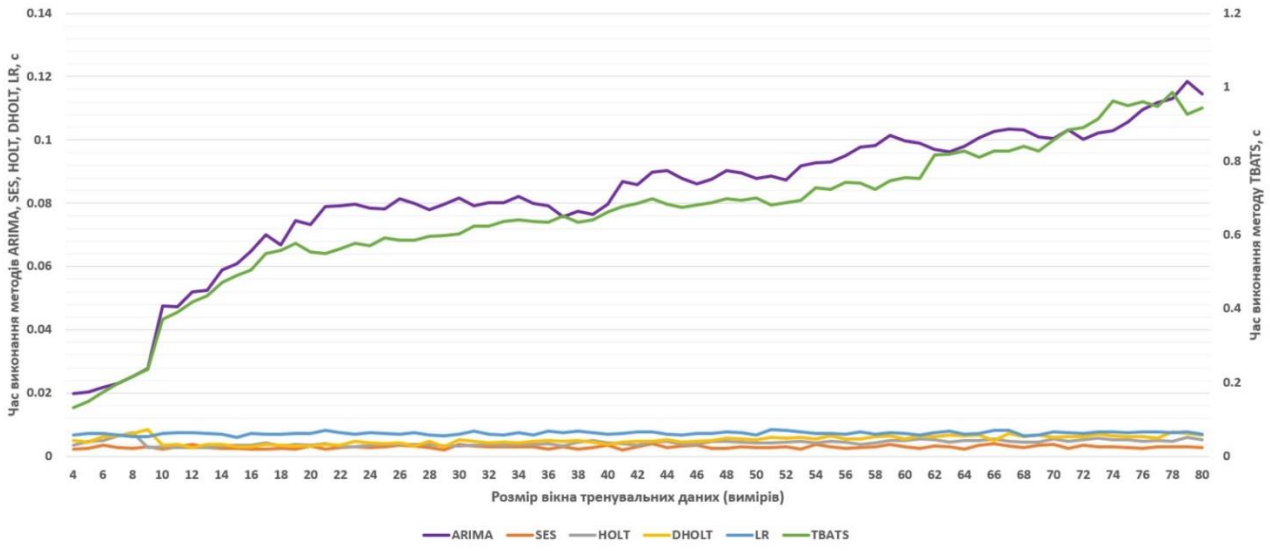


Рис. 8.25. Час роботи альтернативних методів при використанні різних розмірів вікна тренувальних даних для часового ряду BM795

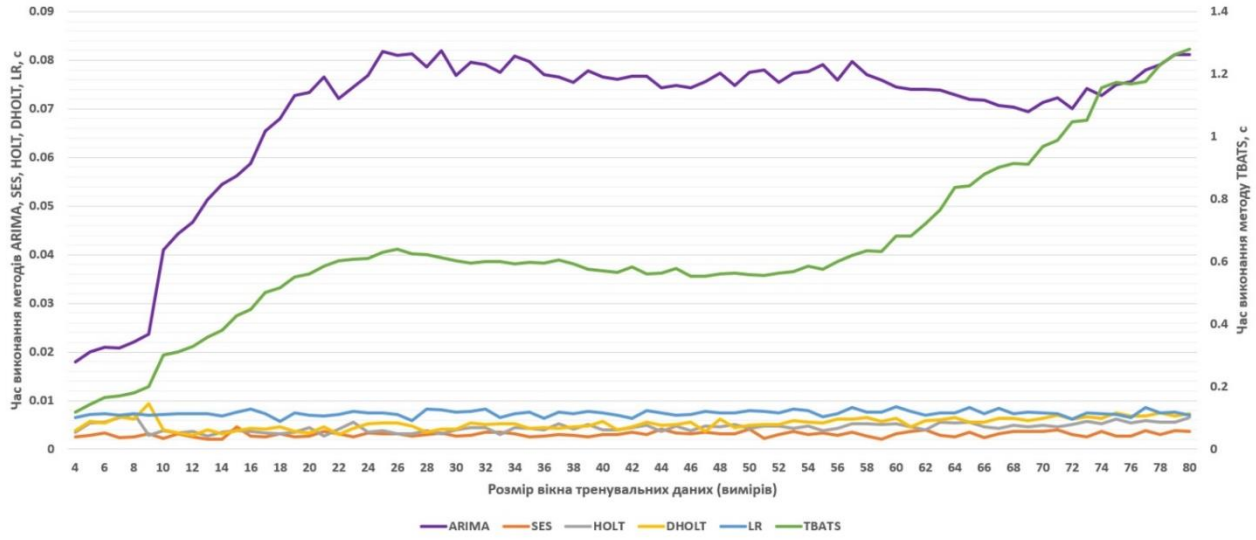


Рис. 8.26. Час роботи альтернативних методів при використанні різних розмірів вікна тренувальних даних для часового ряду BM599

8.5.4. Практичне використання прогнозів

Запропоновані методи прогнозування призначені для отримання прогнозу споживання ресурсу, який вимірюється у відсотках. Тому прогнозоване значення повинно бути в діапазоні $\{0..100\}$. Результати другої групи експериментів з адаптивним методом прогнозування і його комбінованим варіантом виявили наявність прогнозних значень, які є від'ємними, а також значень, які перевищують 100%. Під час управління ресурсами ЦОД з використанням прогнозу в систему управління немає сенсу подавати такі значення. Результати роботи запропонованих методів прогнозування необхідно покращити через заміну отриманих прогнозів згідно рівняння (8.1).

$$\hat{y}_{t+1|t} = \begin{cases} y_{\min}, & \text{якщо } \hat{y}_{t+1|t} < 0, \\ y_{\max}, & \text{якщо } \hat{y}_{t+1|t} > 100. \end{cases}, \tag{8.1}$$

де $y_{\min}$ – мінімальне значення споживання обчислювального ресурсу за деякий останній час, або 0; $y_{\max}$ – максимальне значення споживання обчислювального ресурсу за деякий останній час, або 100.

Якщо застосувати перетворення (8.1) до отриманих результатів, то точність прогнозу підвищується (рис. 8.27 і рис. 8.28).

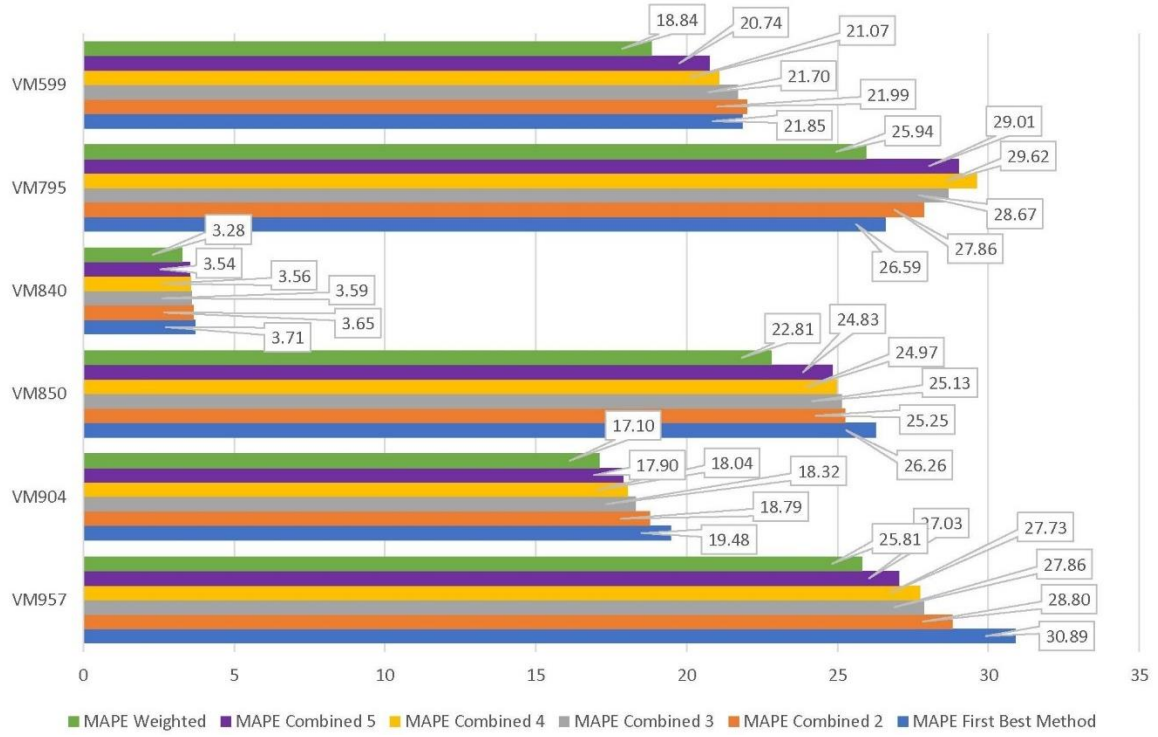


Рис. 8.27 Середня похибка MAPE для часових рядів BM599 – BM957 після застосування трансформації (8.1)

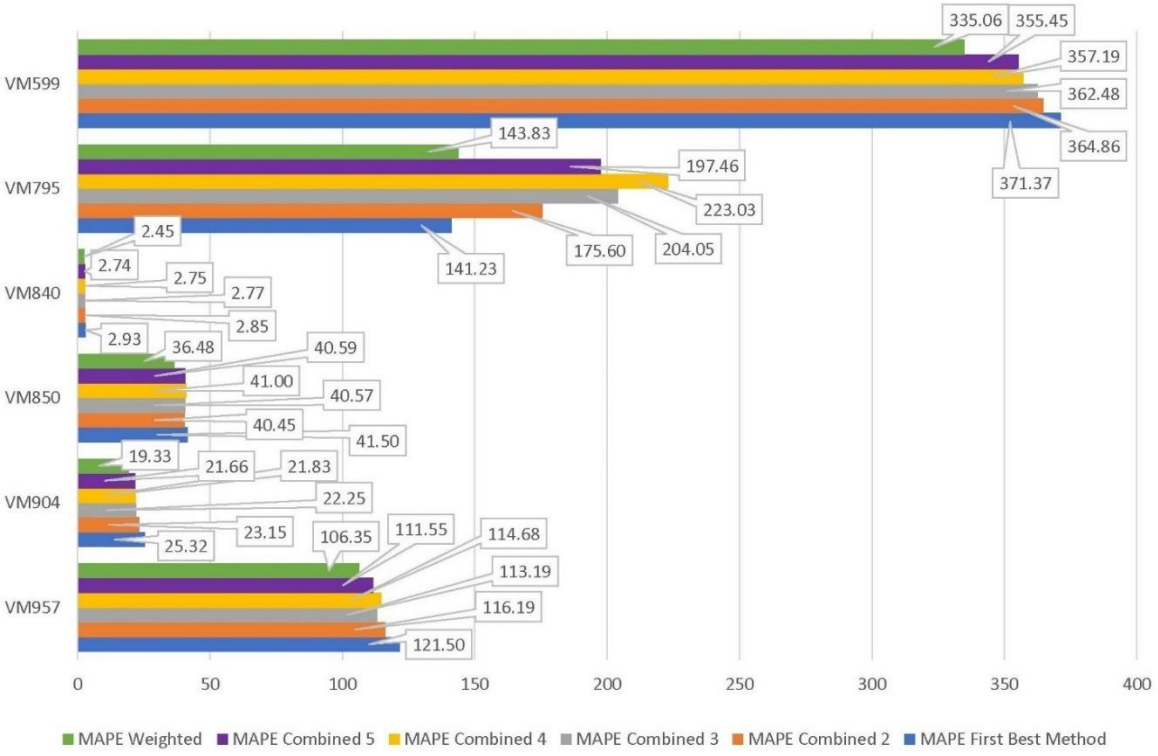


Рис. 8.28. Стандартне відхилення середньої похибка MAPE для часових рядів BM599 – BM957 після застосування трансформації (8.1)

Таким чином, для досліджуваних часових рядів, які вибрані для перевірки роботи адаптивного метода прогнозування, отримана висока точність прогнозу для часового ряду BM840 (MAPE=3.28%), добра точність прогнозів для часових

рядів BM904 (MAPE=17.1%) і BM599 (MAPE=18.8%), задовільна точність прогнозів для часових рядів BM795 (MAPE=25.9%) і BM957 (MAPE=25.8%).

Серед варіантів запропонованого адаптивного методу прогнозування найкращі показники якості прогнозу (середнє значення похибки MAPE, кількість незадовільних прогнозів) отримані за допомогою адаптивного методу зваженого комбінованого прогнозування.

Запропонований адаптивний метод комбінованого прогнозування дозволяє подальшу оптимізацію з метою покращення точності прогнозу через підбір комбінацій $\{a_i, w_j\}$ залежно від характеристик часового ряду, через збільшення кількості попередніх кроків управління для коригування вагових коефіцієнтів та через додавання/виключення альтернативних методів прогнозування.

8.6. Експериментальне дослідження методу управління реплікацією та міжрівневою міграцією даних у сховищі хмарного центру оброблення даних

Для дослідження запропонованої моделі розподіленого дворівневого сховища та методу управління [287, 317, 326, 346] розроблено програмний застосунок з використанням мови програмування Java. Запуск застосунку для симуляції виконувався на комп'ютері з процесором Intel i7-3632QM і об'ємом пам'яті 8 ГБ під управлінням ОС Windows 10 Pro 64bit. Реалізація методу управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД у вигляді діаграми класів застосунку моделювання наведена у додатку А.

Для формування вхідних даних, що подаються на вхід застосунку, використано програмне забезпечення Process Monitor [48]. Process Monitor дає можливість записати перелік процесів, які виконуються і отримують доступ до файлів на дисках при роботі з операційною системою Windows. Збір вхідних даних для симуляції включає в себе декілька етапів. На першому етапі, з використанням Process Monitor, було зібрано дані про використання файлів на дисковому пристрої застосунками Windows звичайного офісного комп'ютера протягом 8 годин. Другий етап. Збереження звіту про роботу Process Monitor у форматі CSV, в який записуються результати моніторингу (рис. 8.29).

Time of Da	Process Na	PID	Operation	Path	Result	Detail	User
01:14.7	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.7	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.7	McUpdate	10052	ReadFile	C:\Prograr	SUCCESS	Offset: 0, I	NT AUTHORITY\SYSTEM
01:14.7	McUpdate	10052	ReadFile	C:\Prograr	SUCCESS	Offset: 36,	NT AUTHORITY\SYSTEM
01:14.7	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.7	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.8	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.8	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.8	McUpdate	10052	ReadFile	C:\Prograr	SUCCESS	Offset: 0, I	NT AUTHORITY\SYSTEM
01:14.8	McUpdate	10052	ReadFile	C:\Prograr	SUCCESS	Offset: 36,	NT AUTHORITY\SYSTEM
01:14.8	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.9	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.9	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:14.9	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM
01:15.0	McUpdate	10052	ReadFile	C:\Prograr	SUCCESS	Offset: 0, I	NT AUTHORITY\SYSTEM
01:15.0	McUpdate	10052	ReadFile	C:\Prograr	SUCCESS	Offset: 36,	NT AUTHORITY\SYSTEM
01:15.1	McUpdate	10052	CreateFile	C:\Prograr	SUCCESS	Desired Ac	NT AUTHORITY\SYSTEM

Рис. 8.29. Фрагмент CSV файлу при експорті з Process Monitor

Третій етап. Формування вхідних даних для дослідження моделі розподіленого дворівневого сховища та якості використання представлених в роботі алгоритмів (рис. 8.30). На вхід кожної ВМ при моделюванні подається така інформація про активність дискової підсистеми: файл та шлях до нього; розмір файлу; чи змінювався його розмір протягом часу роботи Process Monitor; кількість запитів до файлу.

Path	Size	Change Siz	Count
"C:\Windo	0	FALSE	9
"C:\Windo	0	FALSE	22
"C:\Windo	14336	FALSE	2
"C:\Windo	249856	FALSE	394
"C:\Windo	0	FALSE	24
"C:\Windo	444752	FALSE	20
"C:\Windo	636048	FALSE	28
"C:\Windo	65536	FALSE	11
"C:\Windo	2319872	FALSE	91
"C:\Windo	491520	FALSE	24
"C:\Progra	16086	TRUE	1490
"C:\Progra	190	FALSE	13
"C:\Windo	115200	FALSE	2953
"C:\Windo	16896	FALSE	1681
"C:\Progra	19618	FALSE	438
"C:\Progra	25179	FALSE	104
"C:\Progra	32	FALSE	144

Рис. 8.30. Вхідні дані для симуляції дворівневого сховища

Для виконання третього етапу підготовки вхідних даних розроблений окремий застосунок з метою конвертації даних формату Process Monitor у формат застосунку симуляції.

Для моделювання розподіленого дворівневого сховища з реплікацією і дослідження запропонованого методу управління необхідно задати такі початкові дані: кількість ФС, які необхідні для симуляції; кількість пристроїв, які знаходяться на швидкому рівні на кожному з ФС системи; кількість пристроїв, які знаходяться на повільному рівні. Крім того, в конфігураційний файл

симулятора треба додати дані про конфігурацію пристроїв кожного рівня та кількість ВМ на кожному ФС.

Характеристики роботи дискових пристроїв, які розміщуються на швидкому рівні: затримка 1 мс; середній час доступу до файлів 1 мс; пропускна спроможність дискового пристрою 300 МБ/с; об'єм диску 64ГБ. Характеристики роботи дискових пристроїв на повільному рівні сховищ: затримка 6 мс; середній час доступу до файлів 9 мс; пропускна спроможність дискового пристрою 120 МБ/с; об'єм диску 500ГБ.

Симулятор створює введену кількість ФС в системі моделювання, до кожного з ФС підключене локальне сховище. На кожному ФС створюються два рівні пристроїв, на кожному з рівнів створюється задана кількість пристроїв. Після створення ФС генеруються ВМ, які будуть отримувати доступ до сховища згідно параметрам, що записані у вхідних даних. Для кожної ВМ, в три етапи, з використанням Process Monitor, згенеровано окремий вхідний файл з описом характеристик доступу до файлів (частота, розмір, вид операції та ін.).

В експерименті на кожному ФС моделюється робота п'яти ВМ. Відповідно, для роботи із застосунком моделювання використано п'ять CSV файлів, в яких містяться дані про доступ до файлів ОС Windows загальним розміром 8742 МБ. Міграція даних між рівнями сховища одного ФС відбувалась у випадку наявності певної визначеної кількості транзакцій доступу до певного файлу ОС, що приводило до необхідності міграції. При великому навантажені на ФС, коли протягом всього часу моделювання від кожної ВМ до сховища надходять запити на обробку великої кількості даних, міграція допомагає зменшити час очікування даних зі сховищ через збереження часто запитуваних файлів на пристроях швидкого рівня.

При моделюванні фактор реплікації дорівнює трьом (RF=3), тобто існують три копії файлів на трьох різних ФС (вузлах зберігання даних). Метод управління реплікаціями даних дозволяє створити дві копії кожного блоку файлу і розмістити їх на ФС, показники якості яких (коефіцієнти реплікації) є краще. Створення реплік блоків даних на різних вузлах дозволяє завантажувати всі вузли рівномірно, з метою уникнення їх перевантаження. В поточній версії

застосунку моделювання роботи з окремими блоками файлів не виконується. Таким чином, при доступі до файлу він цілком записується на швидкий рівень.

8.6.1. Оцінка результатів моделювання міграцій

Під час запусків застосунку моделювання встановлено, що наявна інтенсивність роботи з файлами з боку ВМ не призвела до вичерпання дискового простору на швидкому рівні сховища. Поріг підтримки вільного місця на кожному з дисків швидкого рівня встановлений у розмірі 100 МБ, виходячи з розміру найбільшого файлу у вхідних даних. Затримка передачі даних між ФС генерувалась випадковим чином в діапазоні 2-8 мс.

Унаслідок моделювання отримані залежності часу доступу до файлу залежно від його розміру. Моделювання виконано для двох конфігурацій сховищ: з дворівневим сховищем і з однорівневим сховищем. Однорівневе сховище моделюється пристроями повільного рівня. Для кожної конфігурації виконано десять запусків моделювання. У результаті, отримані середні показники. Так як запис файлів на дворівневе сховище в більшості випадків відбувається на швидкий рівень, то час запису визначається показниками пристроїв швидкого рівня (в середньому 95 мс) (рис. 8.31).

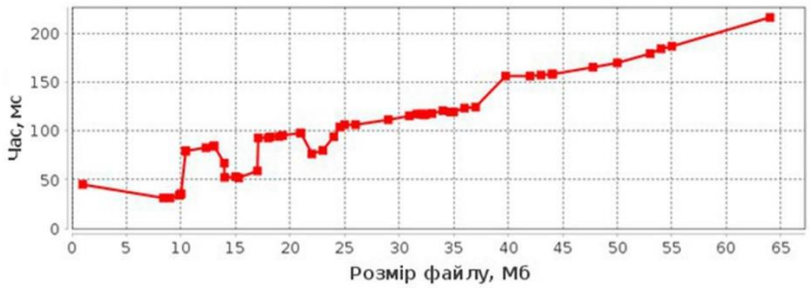


Рис. 8.31. Запис файлів в дворівневе сховище

Виключення може бути тоді, коли створюється новий файл, розмір якого вище, чим вільне місце на одному з пристроїв швидкого рівня. У такому випадку файл створюється на пристрої повільного рівня. При цьому, паралельно, виконується очистка пристроїв швидкого рівня з метою подальшого перенесення цього нового файлу на швидкий рівень (в фоновому режимі). Таким чином, середній час запису такого файлу трохи зростає. При записі файлів в однорівневе сховище витрачається більший час (в середньому 192 мс) (рис. 8.32) через те, що

продуктивність пристроїв повільного рівня нижча за продуктивність пристроїв швидкого рівня.

При роботі дворівневого сховища в режимі читання файлів перша транзакція читання файлу відбувається з пристрою повільного рівня з одночасним записом цього файлу на швидкий рівень. Подальша робота з файлом вже буде відбуватися через доступ до пристроїв швидкого рівня. Виключенням є ситуація, коли перенесення цього файлу на швидкий рівень неможливо через нестачу вільного місця. У такому випадку запускається алгоритм очистки пристроїв швидкого рівня паралельно з доступом до цього файлу на пристрої повільного рівня.

Рис. 8.32. Запис файлів в однорівневе сховище

Коли вільне місце з’являється на пристрої швидкого рівня цей файл мігрує на швидкий рівень і подальша робота з ним відбувається вже з вищою продуктивністю. Таким чином, при зчитуванні файлів з багаторівневого сховища (рис. 8.33) час очікування доступу до файлу є меншим (в середньому 262 мс) через те, що дані знаходяться на швидкому рівні.

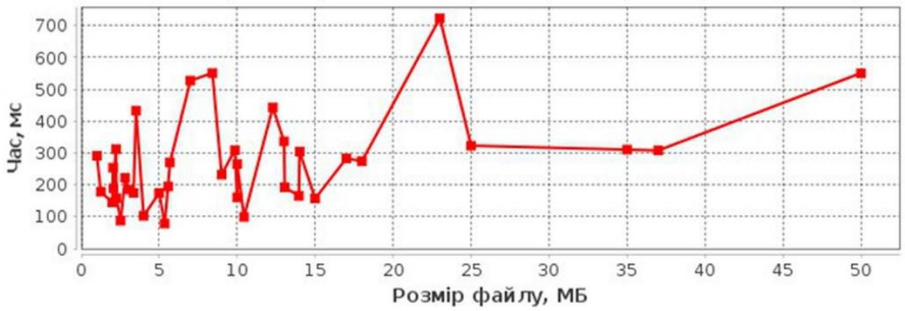


Рис. 8.33. Читання файлів з дворівневого сховища

Таким чином, розглядаючи усереднені значення зчитування і запису деякої кількості файлів з використанням дворівневого сховища та однорівневого, можна зробити висновок, що використання міграції дозволяє зменшити час

очікування доступу до файлів при виконанні одночасних дискових операцій ВМ на ФС.

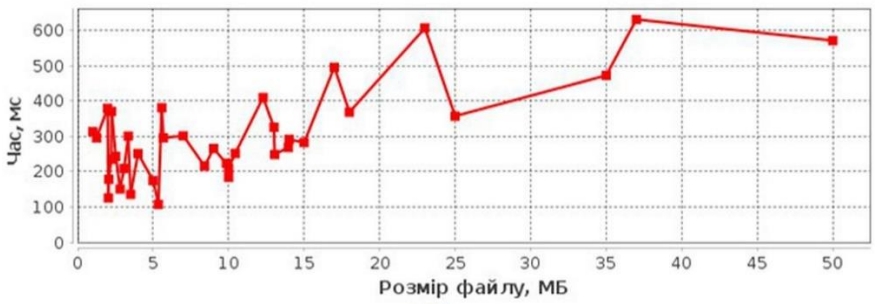


Рис. 8.34. Читання файлів з однорівневого сховища

8.6.2. Оцінка результатів моделювання реплікацій

Моделювання реплікацій виконано з урахуванням додаткового обмеження, при якому всі блоки одного файлу реплікуються тільки на один вузол. На початковому етапі моделювання сховища кожного ФС зберігали обсяг даних, як показано в табл. 8.13. Стандартне відхилення складає 1,18 ГБ.

Табл. 8.13. Зайнятість сховищ ФС даними

	Кількість зайнятого місця на сховищі, МБ
Сервер 1	12375
Сервер 2	10517
Сервер 3	11613
Сервер 4	9219
Сервер 5	11427

Пошук вузлів для розміщення реплік при моделюванні виконувався по порядку, починаючи з сервера 1 до сервера 5. Спочатку виконано пошук серверів для розміщення реплік сервера 1, потім – сервера 2 і т.д. Унаслідок, розподіл реплік файлів серверів відбувся у такий спосіб (табл. 8.14). Стандартне відхилення складає 1,81 ГБ.

Табл. 8.14. Розподіл даних по сховищам ФС (по вузлах)

№ сервера	Кількість зайнятого місця у сховищі, МБ	Кількість реплікацій	Сервери, на які відбувається реплікація
1	35415	2	2,4
2	34319	2	3,5
3	31349	2	1,4
4	33207	2	3,5
5	31163	2	1,2

Таким чином, розглядаючи стандартне відхилення об’єму даних, що розміщені на вузлах до реплікації і після, можна зробити висновок, що запропонований метод реплікації дозволяє рівномірно розмістити репліки даних по вузлах ЦОД не створюючи вузьких місць при створенні нових і модифікації існуючих блоків даних.

Висновки до розділу 8

1. У результаті аналізу трейсів роботи кластера Google можна побачити, що попит на кожний тип ресурсу значно коливається в часі. Ефективність використання електроенергії оцінена через порівняння енергоспоживання активних ФС під час моделювання згідно розроблених в дисертації моделей інтегрованого управління з результатами, отриманими за допомогою алгоритму Power Aware Best Fit Decrease (PABFD) [122]. Розроблений в дисертаційній роботі метод ІУР ІТ-інфраструктури хмарного ЦОД дозволяє розподіляти ВМ на ФС в середньому на 17% ефективніше, ніж PABFD. Після застосування запропонованої моделі управління ємністю в сценаріях моделювання, затримка планування зменшується до 37% у деяких сценаріях (вибірках з трейсу). Таким чином, цей метод орієнтований на планування виконання неоднорідних завдань у гетерогенних середовищах ЦОД.

2. У розділі оцінені показники якості алгоритмів методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу при різних умовах навантаження та порівняні з результатами, отриманими в [194]. Результати вказують на те, що при використанні обох алгоритмів методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу ЦОД споживає майже таку ж кількість енергії, як і алгоритм LR-MMT-1.2, тому що алгоритми методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу використовують LR і MMT політики, але виконують тільки розміщення ВМ відповідно до техніки оптимізації відпалу. При використанні простого алгоритму OIB, метрика *PDM* не змінюється порівняно з алгоритмом LR-MMT-1.2, оскільки більш рівномірне завантаження ФС не впливає безпосередньо на кількість міграцій ВМ. Більш рівномірне завантаження ФС за рахунок застосування

методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу дозволяє зменшити метрику *SLATAH*. Алгоритм OIB з обмеженням міграцій дозволяє зменшити кількість міграцій і, як наслідок, зменшити метрику *PDM*. З іншого боку, алгоритм OIB з обмеженнями дозволяє знизити загальне падіння продуктивності роботи ВМ через міграції, внаслідок чого порушення SLA зменшено. Рівномірне завантаження ФС призводить до зменшення порушень SLA через резервування деяких ресурсів ФС для реагування на зростаючі випадкові вимоги до ресурсів у найближчому майбутньому.

3. Для управління ресурсами хмарного ЦОД на рівні кластера досліджено двостадійний метод управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку. Для перевірки роботи алгоритму з різними методиками та їх порівняння створено декілька наборів даних з однаковою кількістю ФС та завдань за інші періоди часу в наборі даних GCT. Отримані показники ефективності розрізняються не більше ніж на 7%.

Для оцінки запропонованого методу експерименти проведені на вхідних даних з шести фрагментів масивів даних використання кластера Google. Проведено аналіз запропонованих методик нижньої границі та порогу вільних ресурсів. Унаслідок визначено, що методика з порогом вільних ресурсів показала більш ефективні результати при вирішенні задачі консолідації ВМ.

Запропонований метод дозволяє переключити в режим сну в середньому 56% ФС, що потенційно визначені для переключення в режим сну за допомогою верхньої оцінки необхідної ємності ресурсів при врахуванні допустимої кількості міграцій ВМ. Також встановлено, що рекомендована ширина променя в алгоритмі променевого пошуку складає від 5 до 8, залежно від умов роботи методу, обмежень на кількість міграцій ВМ та обмежень на час виконання алгоритму.

4. Проаналізовано метод динамічної консолідації і розміщення ВМ на основі алгоритму навчання з підкріпленням для управління ресурсами хмарних ЦОД. Оцінка продуктивності на основі реальних даних робочого навантаження від Bitbrains [22] демонструє ефективність запропонованого методу. Запропонований метод дозволяє зменшити час порушення SLA, обслуговувати

більше запитів створення нових ВМ, коли кількість ВМ часто змінюється, і зменшувати використання мережі ЦОД через зменшення кількості міграцій. Запропонований в [324] метод покращений через розроблення основного алгоритму, зменшення розміру простору стану, зменшення кількості інтервалів використання ресурсів і кількості атрибутів елемента простору стану і представлений в роботі [330].

Оцінка ефективності методу виконана з урахуванням трьох показників якості: часу порушення SLA, кількості споживаної енергії та кількості міграцій ВМ. Визначені коефіцієнти моделі. Розроблений метод усуває два недоліки політик, запропонованих в [203]: високу затримку при плануванні створення нової ВМ через відсутність доступних ФС, що задовольняють вимогам з боку ВМ; збільшення споживання енергії при перемикання ФС з активного стану в сплячий режим і навпаки та коли кількість ВМ часто змінюється протягом відносно короткого періоду часу.

Розроблений метод дозволяє вимикати достатню кількість ФС, що зменшує загальний час налаштування ФС, і, відповідно, зменшує затримку планування нової ВМ, коли немає активних ФС, які можуть задовольнити запит планування нової ВМ, або міграції ВМ з перевантажених ФС. При моделюванні враховується вплив кількості вхідних параметрів роботи ресурсів, які не враховуються за замовченням в середовищі CloudSim.

5. У розділі досліджено адаптивний двоетапний метод прогнозування споживання обчислювальних ресурсів і адаптивні методи комбінованого прогнозування (усереднений і зважений) з адаптацією параметрів моделі. При дослідженні запропонованих методів прогнозування використані такі альтернативні методи/моделі прогнозування: метод простого експоненційного згладжування, метод Хольта, демпфірований метод тренду, моделі авторегресії інтегрованого ковзного середнього, модель лінійної регресії, метод TBATS. Аналіз та моделювання виконані з використанням статистичних даних Bitbrains [22] на прикладі прогнозування споживання процесорного ресурсу. Якісний аналіз результатів дослідження показав, що точність прогнозу в значній мірі залежить від статистичних характеристик часового ряду. Аналіз також показує,

що прогнозування сплесків практично неможливо, оскільки вони можуть бути викликані збоями ФС і нестаціонарними моделями використання ресурсів. Результати кількісного аналізу показують, що точність прогнозу, отриманого за допомогою спрощеного адаптивного методу, збільшується в середньому на 2.4% - 23.6% залежно від часового ряду. Точність прогнозу, отриманого адаптивним методом зваженого комбінованого прогнозування, збільшується в середньому на 6.6% - 21.2% (залежно від часового ряду) у порівнянні з точністю спрощеного адаптивного методу прогнозування. Найкращий прогноз споживання процесорного ресурсу отриманий за допомогою адаптивного зваженого комбінованого методу прогнозування з трансформацією.

Унаслідок аналізу продуктивності найбільш трудомісткими методами є TBATS і ARIMA. Вони можуть бути використані в якості альтернативних методів прогнозування, коли інтервал вимірювань даних моніторингу становить більше 100 секунд і 10 секунд відповідно.

6. Для дослідження запропонованої моделі та методу управління сховищем розроблено програмний застосунок, що дозволяє моделювати роботу дворівневого і однорівневого сховища з реплікацією. Результати дослідження показують, що використання дворівневих сховищ із запропонованим методом управління призводить до зменшення необхідного об’єму пристроїв швидкого рівня (зниження вартості збереження даних) та зменшення часу очікування завершення транзакцій доступу до файлів при одночасній роботі ВМ зі сховищем ФС. Запропонований метод реплікації дозволяє рівномірно розмістити репліки даних по вузлах ЦОД не створюючи вузьких місць при створенні нових і модифікації існуючих блоків даних. Стандартне відхилення об’ємів даних, що зберігаються на вузлах, зменшилося на 35%.

Аналіз результатів симуляції роботи дворівневого сховища і запропонованого методу управління показав, що час очікування завершення транзакцій доступу до файлів зменшується, що, у порівнянні з однорівневим сховищем, побудованим на повільних пристроях, призводить до підвищення продуктивності роботи дворівневого сховища при одночасній роботі ВМ і

контейнерів. У середньому, при використанні дворівневого сховища запис виконується на 51%, а читання – на 16% швидше.

Подальше дослідження запропонованої моделі і методу управління пов’язане з доробкою застосунка моделювання з метою урахування блочного обміну та підбору вагових коефіцієнтів для досягнення високої продуктивності роботи розподіленого дворівневого сховища.

ВИСНОВКИ

У дисертаційній роботі вирішено науково-практичну проблему забезпечення ефективного функціонування ІТ-інфраструктури хмарного ЦОД через створення методології управління та на її основі відповідної ІТУ із застосуванням розроблених підходів, моделей, алгоритмів і методів, які базуються на методах інтегрованого і ієрархічного управління, штучного інтелекту, моделях і методах прогнозування навантажень і споживання ресурсів ІТ-інфраструктури, що дозволило забезпечити виконання заданих вимог угоди про рівень обслуговування та зниження операційних і капітальних витрат в умовах невизначеності та змінних навантажень. У процесі вирішення поставлених задач в дисертаційній роботі отримані такі наукові результати:

1. Аналіз існуючих підходів, технологій, моделей і методів управління ІТ-інфраструктурою ЦОД провайдера хмарних послуг виявив необхідність розроблення методології управління ІТ-інфраструктурою та на її основі інформаційної технології, моделей і методів з метою досягнення таких показників ефективності, як: зменшення споживання електроенергії, підвищення якості обслуговування споживачів, зменшення капітальних та операційних витрат, зменшення кількості збоїв та простоїв.

2. На основі операторної форми постановки, аналізу і розв’язання задач управління ІТ-інфраструктурою хмарного ЦОД запропоновано методологію управління для реалізації програмно-визначеного підходу в умовах невизначеності і змінних навантажень через використання множини напрацьованих схем реалізацій з можливістю їх розширення, а також комбінування моделей і методів запропонованого комплексу в залежності від початкових і поточних умов функціонування з урахуванням результатів прогнозування.

3. На основі запропонованої методології з використанням комплексу моделей і методів управління розроблено оригінальну інформаційну технологію, яка забезпечує збір, накопичення, оброблення і використання інформації для ефективного управління ІТ-інфраструктурою хмарного ЦОД в умовах змінних навантажень, яка на відміну від відомих враховує суттєві характеристики

хмарних обчислень, гетерогенність ІТ-інфраструктури, нові архітектури організації обчислень, забезпечує адаптацію до змінних навантажень за рахунок прогнозування і може бути застосована для реалізації функцій, підсистем, компонентів та інших складових інформаційної системи управління.

4. Розроблено адаптивний метод комбінованого прогнозування навантаження на обчислювальні ресурси хмарного ЦОД з використанням усередненого, зваженого та оптимізаційного комбінування оцінок прогнозів, обчислених за альтернативними методами прогнозування та з адаптацією розміру навчальної вибірки, який відрізняється обчисленням та використанням певних типів комбінованих прогнозів (усередненого та зваженого) в реальному часі, отриманих альтернативними моделями і методами, що дозволяє збільшити точність прогнозу в середньому на 6.6% – 21.2% в залежності від часового ряду та застосувати його в умовах провайдера хмарних послуг для прогнозування відомих видів змішаних навантажень в ІТ-інфраструктурі.

5. Розроблено метод інтегрованого управління фізичними і віртуальними машинами в ІТ-інфраструктурі хмарного ЦОД на основі динамічної моделі станів із застосуванням стохастичного пошуку для виконання міграцій ВМ, вивільнення, увімкнення і вимкнення ФС та прогнозування для адаптації до змін навантаження з метою досягнення сталого режиму роботи згідно визначених критеріїв. Запропонований метод інтегрованого управління дозволяє розподіляти ВМ на ФС в середньому на 17% ефективніше, ніж за допомогою відомого аналогу, що дозволяє застосувати його при розробленні модулів управління в складі ПЗ з відкритим кодом, що застосовується для побудови хмарних ЦОД.

6. Удосконалено комплекс алгоритмів і методів стохастичного пошуку через розроблення нових методів управління ресурсами ІТ-інфраструктури хмарного ЦОД в складі запропонованої методології, а саме методу рівномірної консолідації ВМ з використанням ідеї імітації відпалу, двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку, методу динамічної консолідації і розміщення ВМ на основі алгоритму навчання з підкріпленням, що відрізняються врахуванням методик вибору станів системи

на основі нових метрик, врахуванням достатньої кількості видів ресурсів і дискретності вимірів, а також особливостей функціонування ІТ-інфраструктури хмарного ЦОД в умовах віртуалізації із застосуванням програмно-визначеного управління, що дозволяє застосувати їх при розробленні і модернізації існуючих комплексів управління ресурсами ІТ-інфраструктури хмарного ЦОД.

7. Розроблено архітектуру та структурно-функціональну модель багаторівневої ієрархічної програмно-визначеної СУІ хмарного ЦОД, яка за рахунок поєднання стратегічного і оперативного управління з автоматичним керуванням, а також надання програмно-визначених властивостей дозволяє застосувати її при управлінні ресурсами і навантаженням ІТ-інфраструктури хмарного ЦОД з вибором стратегій, плануванням і управлінням їх реалізацією, автоматичним керуванням з урахуванням структури системи, історичних даних її функціонування та дотриманням заданих показників якості угоди про рівень обслуговування.

8. Декомпозиційно-компенсаційний підхід до управління ІТ-інфраструктурою отримав розвиток через надання адаптивності, багаторівневості та програмно-визначених властивостей, що дозволило реалізувати ефективне управління ресурсами ІТ-інфраструктури хмарного ЦОД з переходом від вибору стратегій до планування і управління їх реалізацією з урахуванням структури системи. Запропонований декомпозиційно-компенсаційний підхід реалізовано для управління інфраструктурою IoT на основі використання мікрохмари через виділення рівнів координації послуг, планування ресурсів та управління рівнем обслуговування в інтегрованій системі управління, що дозволило забезпечити задану якість ІТ-послуг при раціональному використанні ресурсів.

9. Розроблено метод управління реплікацією та міжрівневою міграцією даних сховища хмарного ЦОД у складі інформаційної технології управління, який використовує кешування за розміром файлу і за кількістю транзакцій доступу до файлу для управління міграцією і використовує кількість транзакцій доступу до блоків даних, об’єм вільного місця та затримку передачі даних між вузлами зберігання даних для управління реплікацією, що дозволяє у середньому

виконувати запис на 51%, а читання – на 16% швидше і застосувати його в умовах хмарного ЦОД для підвищення рівня надійності збереження даних, відмовостійкості та продуктивності їх оброблення.

10. Для оцінювання стану хмарного ЦОД засобами інформаційної технології управління запропоновано і експериментально обґрунтовано ефективність використання нових метрик, зокрема миттєвого і середнього коефіцієнтів життєздатності віртуальної машини, індикатору дисбалансу фізичного сервера, коефіцієнту відношення необхідних ресурсів до середнього об’єму наявних ресурсів, порогу вільних ресурсів та метрики ємності ЦОД, що відрізняються більш чітким оцінюванням стану і динаміки змін ресурсів хмарного ЦОД і дозволяє ефективніше визначати змінні, обмеження і критерії на множині моделей і методів управління нижнього рівня.

11. Розроблено метод управління потужністю хмарного ЦОД у складі інформаційної технології управління на основі динамічної моделі його станів, який використовує запропоновані метрики, враховує гетерогенність ФС і визначає тип і кількість ФС, що треба увімкнути для обслуговування прогнозованого навантаження, що дає можливість провайдеру завчасно автоматично вмикати необхідну кількість ФС потрібної конфігурації і зменшити затримку розгортання сервісів користувача у хмарі на 18%.

12. Отримані в дисертаційній роботі результати досліджень використані при модернізації систем управління функціонуванням інформаційно-телекомунікаційної інфраструктури в компаніях-членах Асоціації «ТЕЛАС», при розробленні системи управління функціонуванням інформаційно-телекомунікаційної інфраструктури ТОВ «АМ ІНТЕГРАТОР ГРУП» та використані при модернізації системи управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО». Впровадження результатів досліджень дозволило на 28% скоротити операційні витрати на управління ІТ-інфраструктурою без порушень SLA; зменшити споживання електроенергії на 17% в середовищі з гомогенними конфігураціями ФС; скоротити в середньому на 18% кількість ФС, що обслуговують навантаження клієнтів; зменшити кількість порушень SLA на 27% при наданні ІТ-послуг; скоротити витрати на експлуатацію серверного парку на

19% при забезпеченні виконання заданих вимог угоди про рівень обслуговування.

13. Теоретичні і практичні результати дисертаційної роботи склали основу нових спецкурсів, що викладаються автором на кафедрі автоматизованих систем обробки інформації і управління Національного технічного університету України “Київський політехнічний інститут імені Ігоря Сікорського”: “Технології віртуалізації та хмарних обчислень”, “Сучасні технології розроблення програмного забезпечення”, “Інтелектуальні системи управління технічними пристроями”, “Програмування інтернету речей”, “Методи та системи штучного інтелекту”, “Обробка надвеликих масивів інформації”.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., and Warfield, A. (2003), “Xen and the art of virtualization”, In *ACM SIGOPS operating systems review*, vol. 37, no. 5, pp. 164-177.
2. Chen, G., He, W., Liu, J., Nath, S., Rigas, L., Xiao, L., and Zhao, F. (2008), “Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services”, In *NSDI*, Vol. 8, pp. 337-350.
3. Padala, P., Hou, K. Y., Shin, K. G., Zhu, X., Uysal, M., Wang, Z., and Merchant, A. (2009), “Automated control of multiple virtualized resources”, *Proc. of the ACM European conference on Computer systems (EuroSys’09)*.
4. Gross, G., and Galiana, F. D. (1987), “Short-term load forecasting”, *Proceedings of the IEEE*, *75*(12), pp. 1558–1573.
5. Xiao, Z., Song, W., and Chen, Q. (2013), “Dynamic resource allocation using virtual machines for cloud computing environment”, *Parallel and Distributed Systems, IEEE Transactions* on, vol. 24, no. 6, pp. 1107–1117.
6. Xue, J., Yan, F., Birke, R., Chen, L. Y., Scherer, T., and Smirni, E. (2015), “PRACTISE: Robust prediction of data center time series”, *11th International Conference on Network and Service Management (CNSM)*, pp. 126–134.
7. Farahnakian, F., Liljeberg, P., and Plosila, J. (2013), “LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers”, *39th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 357–364.
8. Islam, S., Keung, J., Lee, K., and Liu, A. (2012), “Empirical prediction models for adaptive resource provisioning in the cloud”, *Future Generation Computer Systems*, vol. 28, no. 1, pp. 155–162.
9. Dabbagh, M., Hamdaoui, B., Guizani, M., and Rayes, A. (2015), “Energy-efficient resource allocation and provisioning framework for cloud data centers”, *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 377–391.
10. Naseera, S., Rajini, G. K., Prabha, N. A., and Abhishek, G. (2015), “A comparative study on CPU load predictions in a computational grid using artificial neural network algorithms”, *Indian Journal of Science and Technology*, vol. 8, no. 35.
11. Naseera, S., Rajini, G. K., and Reddy, P. S. K. (2016), “Host CPU Load Prediction Using Statistical Algorithms a comparative study”, *International Journal of Computer Technology and Applications,* 9 (12). pp. 5577–5582.
12. Dinda, P. A. (2006), “Design, implementation, and performance of an extensible toolkit for resource prediction in distributed systems”, *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 2, pp. 160–173.
13. Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015), “Time series analysis: forecasting and control”, 5th ed. Hoboken, NJ, USA: John Wiley & Sons.
14. Montgomery, Douglas C., Elizabeth A. Peck, and G. Geoffrey Vining, (2015). “Introduction to linear regression analysis”, John Wiley & Sons.

15. Park, K., and Pai, V. S. (2006), “CoMon: a mostly-scalable monitoring system for PlanetLab”, *ACM SIGOPS Operating Systems Review*, pp. 65-47.
16. Telenyk S., Zharikov E., Rolik O. Architecture and conceptual bases of cloud IT infrastructure management // Advances in Intelligent Systems and Computing. – Springer, Cham, 2017. – Vol. 512. – C. 41-62. http://dx.doi.org/10.1007/978-3-319-45991-2_4.
17. R Core Team (2018), “R: A language and environment for statistical computing”, *R Foundation for Statistical Computing*, Vienna, Austria. URL https://www.R-project.org/.
18. Jorgensen, M. (1995), “Experience with the accuracy of software maintenance task effort prediction models”, *IEEE Transactions on Software Engineering*, vol. 21, pp. 674–681.
19. Hyndman, R., Koehler, A. B., Ord, J. K., and Snyder, R. D. (2008). “Forecasting with exponential smoothing: the state space approach”, *Springer Science & Business Media*.
20. Holt C. C. (2004), “Forecasting seasonals and trends by exponentially weighted moving averages”, *International journal of forecasting*, vol. 20, no. 1, pp. 5-10.
21. Gardner Jr E. S., McKenzie E. D. (1985), “Forecasting trends in time series”, *Management Science*, vol. 31, no. 10, pp. 1237-1246.
22. GWA-T-12 Bitbrains [Online] Available from: http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains [Accessed September 12, 2018].
23. Hyndman R. J., and Khandakar Y. (2008), “Automatic time series forecasting: The forecast package for R”, *Journal of Statistical Software*, 27(1), pp. 1–22. Retrieved from https://www.jstatsoft.org/article/view/v027i03
24. Shen S., van Beek V., and Iosup A. (2015), “Statistical characterization of business-critical workloads hosted in cloud datacenters”, *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pp. 465–474.
25. Hyndman R., Bergmeir C., Caceres G., Chhay L., O'Hara-Wild M., Petropoulos F., Razbash S., Wang E. and Yasmeen F. (2018), “_forecast: Forecasting functions for time series and linear models_”, *R package version 8.3*, <URL: http://pkg.robjhyndman.com/forecast>.
26. Бідюк П. І., Меняйленко О. С., Половцев О. В. Методи прогнозування //Луганськ: Альма-матер. – 2008. – Т. 1. – С. 308.
27. Hyndman R. J., Athanasopoulos G. Forecasting: principles and practice. – OTexts, 2018.
28. De Livera A. M., Hyndman R. J., Snyder R. D. Forecasting time series with complex seasonal patterns using exponential smoothing //Journal of the American Statistical Association. – 2011. – Т. 106. – №. 496. – С. 1513-1527.
29. R-3.4.4 for Windows (32/64 bit) https://cran.r-project.org/bin/windows/base/old/3.4.4/
30. Bates, J. M., & Granger, C. W. J. (1969). The combination of forecasts. Operational Research Quarterly, 20(4), 451–468. https://doi.org/10.1057/jors.1969.103

31. Top 10 Digital Transformation Trends For 2019 Framework [Online] – Available from: https://www.forbes.com/sites/danielnewman/2018/09/11/top-10-digital-transformation-trends-for-2019/#17d55d1e3c30.
32. ZFS L2ARC [Online] – Available from: http://137.254.16.27/brendan/entry/test.
33. Chen, F., Koufaty, D. A., & Zhang, X. (2011, May). Hystor: making the best use of solid state drives in high performance storage systems. In Proceedings of the international conference on Supercomputing (pp. 22-32). ACM.
34. Guerra, J., Pucha, H., Glider, J. S., Belluomini, W., & Rangaswami, R. (2011, February). Cost Effective Storage using Extent Based Dynamic Tiering. In FAST (11). pp. 1-14.
35. Arteaga, D., & Zhao, M. (2014, June). Client-side flash caching for cloud systems. In Proceedings of International Conference on Systems and Storage (pp. 1-11). ACM.
36. NVM Express [Online] – Available from: https://nvmexpress.org/.
37. Ethernet Storage Fabric – Part 1 [Online] – Available from: http://www.mellanox.com/blog/2018/05/ethernet-storage-fabric-part-1/.
38. Top 10 storage trends for 2018 and why you should care [Online] – Available from: https://community.hpe.com/t5/Around-the-Storage-Block/Top-10-storage-trends-for-2018-and-why-you-should-care/ba-p/6995176#.W7DrSXnWhmA.
39. Yang, Z., Hoseinzadeh, M., Andrews, A., Mayers, C., Evans, D. T., Bolt, R. T., ... & Swanson, S. (2017, December). AutoTiering: automatic data placement manager in multi-tier all-flash datacenter. In Performance Computing and Communications Conference (IPCCC), 2017 IEEE 36th International (pp. 1-8). IEEE.
40. Bodanyuk M.E., Karnaukhov O.K., Rolik O.I., Telenyk S.F. (2013). Management of data storage systems Electronics and Communications. (5-76). pp. 81–90. (in Ukrainian).
41. Ryu, J., Lee, D., Han, C., Shin, H., & Kang, K. (2016). File-System-Level Storage Tiering for Faster Application Launches on Logical Hybrid Disks. IEEE Access, 4, 3688-3696.
42. Kakoulli, E., & Herodotou, H. (2017, May). OctopusFS: A distributed file system with tiered storage management. In Proceedings of the 2017 ACM International Conference on Management of Data (pp. 65-78). ACM.
43. Intel IOMeter [Online] – Available from: http://www.iometer.org.
44. FIO: Flexible I/O Tester [Online] – Available from: http://linux.die.net/man/1/fio.
45. Borthakur, D. (2007). The hadoop distributed file system: Architecture and design. Hadoop Project Website, 11(2007), 21.
46. White, T. (2012). Hadoop: The definitive guide. " O'Reilly Media, Inc.".
47. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., ... & Stoica, I. (2012, April). Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (pp. 2-2). USENIX Association.
48. Process Monitor [Online] – Available from: https://docs.microsoft.com/en-us/sysinternals/downloads/procmon.

49. Welford B. P. Note on a method for calculating corrected sums of squares and products //Technometrics. – 1962. – Т. 4. – №. 3. – С. 419-420.
50. G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, “Dynamo: amazon’s highly available key-value store,” in ACM SIGOPS operating systems review, 2007, vol. 41, no. 6, pp. 205–220.
51. Ghemawat S., Gobioff H., & Leung, S.-T.. The Google File System. SOSP '03: Proceedings of the nineteenth ACM Symposium on Operating Systems Principles Bolton Landing, NY: ACM. – 2003. – pp. 29–43.
52. Calder, B., Wang, J., Ogus, A., Nilakantan, N., Skjolsvold, A., McKelvie, S., Xu, Y., Srivastav, S., Wu, J., Simitci, H., etal. “Windows Azure Storage: a highly available cloud storage service with strong consistency,” In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles (2011), ACM, pp. 143–157.
53. Weil, S. A., Brandt, S. A., Miller, E. L., Long, D. D., & Maltzahn, C. (2006, November). Ceph: A scalable, high-performance distributed file system. In Proceedings of the 7th symposium on Operating systems design and implementation (pp. 307-320). USENIX Association.
54. Tanenbaum, A. S., & Van Steen, M. (2007). Distributed systems: principles and paradigms. Prentice-Hall.
55. B. A. Milani and N. J. Navimipour, “A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions,” Journal of Network and Computer Applications, vol. 64, pp. 229–238, 2016.
56. Ibrahim, I. A., Dai, W., & Bassiouni, M.: Intelligent Data Placement Mechanism for Replicas Distribution in Cloud Storage Systems. In Proceedings of the IEEE International Conference on Smart Cloud (SmartCloud), pp. 134-139. (2016, November).
57. Zhao, Y., Li, C., Li, L., & Zhang, P.: Dynamic replica creation strategy based on file heat and node load in hybrid cloud. In Proceedings of the 19th International Conference on Advanced Communication Technology (ICACT), pp. 213-220. (2017, February).
58. Mansouri, N.: Adaptive data replication strategy in cloud computing for performance improvement. Frontiers of Computer Science, Vol. 10, No. 5, pp. 925-935. (2016).
59. Gonzalez, Luis Miguel Vaquero, Luis Rodero-Merino, Juan Cáceres and Maik Lindner, “A break in the clouds: towards a cloud definition.” Computer Communication Review 39, 2008, pp. 50–55.
60. P. Mell and T. Grance, “The NIST definition of cloud computing,” Gaithersburg, USA, Special Publication 800-145, vol. 800, no. 145, 2011, p. 7.
61. Whyte, L. L., Wilson, A. G., & Wilson, D. “Hierarchical structures,” New York, NY (USA): Elsevier Scientific Publishing, 1969, 322 p.
62. Pattee, H. H. “Hierarchy theory: the challenge of complex systems,” (No. 575.1 H5), 1973.
63. A. Nadjar, S. Abrishami and H. Deldari, "Hierarchical VM scheduling to improve energy and performance efficiency in IaaS Cloud data centers," 2015 5th International

Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, 2015, pp. 131-136.
64. N. Kandasamy, S. Abdelwahed and M. Khandekar, "A Hierarchical Optimization Framework for Autonomic Performance Management of Distributed Computing Systems," 26th IEEE International Conference on Distributed Computing Systems (ICDCS'06), 2006, pp. 9-9.
65. G. Keller, M. Tighe, H. Lutfiyya and M. Bauer, "A hierarchical, topology-aware approach to dynamic data centre management," 2014 IEEE Network Operations and Management Symposium (NOMS), Krakow, 2014, pp. 1-7.
66. A. R. Hummaida, N. W. Paton and R. Sakellariou, "SHDF - A Scalable Hierarchical Distributed Framework for Data Centre Management," 2017 16th International Symposium on Parallel and Distributed Computing (ISPDC), Innsbruck, 2017, pp. 102-111.
67. X. Wang, M. Chen, C. Lefurgy and T. W. Keller, "SHIP: A Scalable Hierarchical Power Control Architecture for Large-Scale Data Centers," in IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 1, pp. 168-176, Jan. 2012.
68. H. Moens and F. De Turck, "A scalable approach for structuring large-scale hierarchical cloud management systems," Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013), Zurich, 2013, pp. 1-8.
69. Ролик А.И. Декомпозиционно-компенсационный подход к управлению уровнем услуг в корпоративных ИТ-инфраструктурах / А.И. Ролик // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+, – 2013. – № 58. – С. 78–88.
70. Mesarovic, Mihajlo D., Donald Macko, and Yasuhiko Takahara. Theory of hierarchical, multilevel, systems. Vol. 68. Elsevier, 2000.
71. Moiseev, N. N., "Element of the optimal systems theory," Nauka, 1974.
72. Lunze, Jan, “Feedback control of large-scale systems,” New York: Prentice Hall, 1992.
73. A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, “Above the clouds: A berkeley view of cloud computing,” Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, Rep.UCB/EECS, vol. 28, p. 13, 2009.
74. Fonseca N., Boutaba R. “Cloud services, networking, and management,” John Wiley & Sons, 2015.
75. K. Morris, “Infrastructure As Code: Managing Servers in the Cloud,” O’Reilly Media, 2016.
76. H. Lu, M. Shtern, B. Simmons, M. Smit and M. Litoiu, "Pattern-Based Deployment Service for Next Generation Clouds," 2013 IEEE Ninth World Congress on Services, Santa Clara, CA, 2013, pp. 464-471.
77. U. Breitenb¨ucher, T. Binz, O. Kopp, F. Leymann, and J. Wettinger, “Integrated cloud application provisioning: Interconnecting service-centric and script-centric

management technologies,” in OTM 2013 Conferences On the Move to Meaningful Internet Systems, Springer, 2013, pp. 130–148.
78. Apache Friends, https://www.apachefriends.org/about.html
79. C. Pahl, A. Brogi, J. Soldani and P. Jamshidi, "Cloud Container Technologies: a State-of-the-Art Review," in IEEE Transactions on Cloud Computing. doi: 10.1109/TCC.2017.2702586.
80. Andrikopoulos, Vasilios, et al. "How to adapt applications for the cloud environment," Computing vol. 95, no. 6, 2013, pp. 493-535.
81. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” Future Generation computer systems, vol. 25, no. 6, pp. 599–616, 2009.
82. Li, C.S., Brech, B.L., Crowder, S., Dias, D.M., Franke, H., Hogstrom, M., Lindquist, D., Pacifici, G., Pappe, S., Rajaraman, B. and Rao, J.: Software defined environments: An introduction. IBM Journal of Research and Development, Vol. 58, No.2/3, pp.1–15. (2014).
83. R. Buyya, R. N. Calheiros, J. Son, A. V. Dastjerdi, and Y. Yoon, “Software-defined cloud computing: Architectural elements and open challenges,” In Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014, pp. 1–12.
84. 2019. Amazon EC2. https://aws.amazon.com/ec2. (2019).
85. 2019. Google Cloud Platform. https://cloud.google.com/compute/. (2019).
86. Sizes for Windows virtual machines in Azure https://docs.microsoft.com/en-us/azure/virtual-machines/windows/sizes
87. J. Almeida, V. Almeida, D. Ardagna,´I. Cunha, C. Francalanci, and M. Trubian, “Joint admission control and resource allocation in virtualized servers,” Journal of Parallel and Distributed Computing, vol. 70, pp. 344–362, Apr 2010.
88. B. Addis, D. Ardagna, B. Panicucci, M. S. Squillante, and L. Zhang, “A hierarchical approach for the resource management of very large cloud platforms,” Dependable and Secure Computing, IEEE Transactions on, vol. 10, no. 5, pp. 253–272, 2013.
89. Z. Wan and P. Wang, “A Survey and Taxonomy of Cloud Migration,” in International Conference on Service Sciences, 2014, pp. 175–180.
90. Z. Wan, L. Duan, and P. Wang, “Cloud Migration: Layer Partition and Integration,” in Edge Computing (EDGE), 2017 IEEE International Conference on, 2017, pp. 150–157.
91. M. F. Gholami, F. Daneshgar, G. Beydoun, and F. Rabhi, “Key challenges during legacy software system migration to cloud computing platforms—an empirical study,” 2017.
92. Feitelson, D. G. (2015). Workload modeling for computer systems performance evaluation. Cambridge University Press.
93. Cetinski, K., & Juric, M. B. (2015). AME-WPC: Advanced model for efficient workload prediction in the cloud. Journal of Network and Computer Applications, 55, 191-201.

94. Iqbal, W., Erradi, A., & Mahmood, A. (2018). Dynamic workload patterns prediction for proactive auto-scaling of web applications. Journal of Network and Computer Applications, 124, 94-107.
95. Liu, C., Liu, C., Shang, Y., Chen, S., Cheng, B., & Chen, J. (2017). An adaptive prediction approach based on workload pattern discrimination in the cloud. Journal of Network and Computer Applications, 80, 35-44.
96. Armstrong, J. S. (Ed.). (2001). Principles of forecasting: a handbook for researchers and practitioners (Vol. 30). Springer Science & Business Media.
97. Wallis, K. F. (2011). Combining forecasts–forty years later. Applied Financial Economics, 21(1-2), 33-41.
98. Nowotarski, J., Liu, B., Weron, R., & Hong, T. (2016). Improving short term load forecast accuracy via combining sister forecasts. Energy, 98, 40-49.
99. Che, J. (2015). Optimal sub-models selection algorithm for combination forecasting model. Neurocomputing, 151, 364-375.
100. Amiri, M., & Mohammad-Khanli, L. (2017). Survey on prediction models of applications for resources provisioning in cloud. Journal of Network and Computer Applications, 82, 93-113.
101. Ivakhnenko, A. G. (1968). The group method of data handling (GMDH). Automation, 3, 57-83.
102. Clemen, R. (1989). Combining forecasts: A review and annotated bibliography with discussion. International Journal of Forecasting, 5, 559–608. https://doi.org/10.1016/0169-2070(89)90012-5
103. Gartner, Inc., The Future of the Data Center in the Cloud Era, Gartner Research Document, September 2016.
104. Magic Quadrant for Cloud Infrastructure as a Service, Worldwide. https://www.gartner.com/doc/reprints?id=1-2G2O5FC&ct=150519
105. Amazon EC2 Instance Types, https://aws.amazon.com/ec2/instance-types/?nc1=h_ls
106. C. Reiss, J. Wilkes, and J. L. Hellerstein, “Google cluster-usage traces: format+schema,” Google Inc., Mountain View, CA, USA, Technical Report, 2011.
107. R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, “A survey on virtual machine migration and server consolidation frameworks for cloud data centers,” Journal of Network and Computer Applications, vol. 52, pp. 11–25, 2015.
108. Madni S. H. H., Latiff M. S. A., Coulibaly Y. Resource scheduling for infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities/ Journal of Network and Computer Applications. – 2016. – vol. 68. – pp. 173–200.
109. D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, “Power and performance management of virtualized computing environments via lookahead control,” in Proceedings of the International Conference on Autonomic Computing (ICAC), 2008. D. Kusic, J.O. Kephart, J.E. Hanson [et al.] // Proc. of the International Conference on Autonomic Computing (ICAC). – Chicago, 2008. – P. 3 – 12.

110. N. Roy, A. Dubey, and A. Gokhale, “Efficient autoscaling in the cloud using predictive models for workload forecasting,” in Proc. of IEEE International Conference on Cloud Computing (CLOUD), 2011, pp. 500–507.
111. Q. Zhang, M. F. Zhani, S. Zhang, Q. Zhu, R. Boutaba, and J. L. Hellerstein, “Dynamic energy-aware capacity provisioning for cloud computing environments,” in Proceedings of the 9th IEEE/ACM International Conference on Autonomic Computing, 2012, pp. 145–154.
112. Q. Zhang, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, “Dynamic heterogeneity-aware resource provisioning in the cloud,” IEEE transactions on cloud computing, vol. 2, no. 1, pp. 14–28, 2014.
113. B. Guenter, N. Jain, and C. Williams, “Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning,” in Proc. of IEEE INFOCOM, 2011, Shanghai, China.
114. A. Beloglazov and R. Buyya, “Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints,” IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 7, pp. 1366–1379, 2013.
115. S. Garg, S. Sundaram, H. D. Patel, “Robust heterogeneous data center design: a principled approach,” SIGMETRICS Performance Evaluation Review, vol. 39, pp. 8-30, 2011.
116. W. Liu, H. Li, W. Du, F. Shi, “Energy-aware Task Clustering Scheduling algorithm for heterogeneous clusters,” in Proc. of IEEE/ACM International Conference on Green Computing and Communications (ICGCC), August 2011, Chengdu, China.
117. Minas L, Ellison B. “Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers,” Intel Press, 2009.
118. Fan X, Weber WD, Barroso LA. “Power provisioning for a warehouse-sized computer,” in. Proc. of the 34th Annual International Symposium on Computer Architecture (ISCA 2007), ACM New York, NY, USA, 2007, pp. 13– 23.
119. H. Liu, H. Jin, C.-Z. Xu, and X. Liao, “Performance and energy modeling for live migration of virtual machines,” Cluster computing, vol. 16, no. 2, pp. 249–264, 2013.
120. Q. Zheng and B. Veeravalli, “Utilization-Based Pricing for Power Management and Profit Optimization in Data Centers,” J. Parallel and Distributed Computing, vol. 72, no. 1, pp. 27-34, 2011.
121. Hammersley John, Monte Carlo methods, Springer Science & Business Media, 2013.
122. A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing,” *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, 2012.
123. ENERGY STAR Certified Enterprise Servers. https://data.energystar.gov/Active-Specifications/ENERGY-STAR-Certified-Enterprise-Servers/46gm-kmbv/data
124. RightScale 2018 State of the Cloud Report, https://www.rightscale.com/lp/state-of-the-cloud

125. Detecting bottlenecks in a virtualized environment, https://docs.microsoft.com/en-us/windows-server/administration/performance-tuning/role/hyper-v-server/detecting-virtualized-environment-bottlenecks
126. SPECpower_ssj 2008, http://spec.org/power_ssj2008/
127. Foster, I., Zhao, Y., Raicu, I. and Lu, S.: Cloud computing and grid computing 360-degree compared. In: Grid Computing Environments Workshop, GCE'08, pp.1–10. (2008).
128. Agarwal, S., Yadav, S. and Yadav, A.K.: An Efficient Architecture and Algorithm for Resource Provisioning in Fog Computing. International Journal of Information Engineering and Electronic Business (IJIEEB), Vol.8, No.1, pp.48-61. (2016).
129. Barroso, L.A., Clidaras, J. and Hölzle, U.: The datacenter as a computer: An introduction to the design of warehouse-scale machines. Synthesis lectures on computer architecture, Vol.8, No.3, pp.1–154. (2013).
130. Barroso, L. Warehouse-scale Computers. Invited talk at the USENIX Annual Technical Conference. Santa Clara, CA (2007).
131. Open Network Foundation, "Software-defined networking: The new norm for networks," 2012. [Online] Available from: https://www.opennetworking.org/images/stories/ downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf [Accessed Feb 20, 2016].
132. DMTF DSP-IS0501, Software Defined Data Center (SDDC) Definition. (2015).
133. The Software-Defined Data Center, https://www.vmware.com/solutions/software-defined-datacenter.html, last accessed 2017/12/13.
134. Jararweh Y., Al-Ayyoub M., Benkhelifa E., Vouk M., Rindos A. Software defined cloud: Survey, system and evaluation/ Future Generation Computer Systems. – 2015. – vol. 58, – pp. 56–74.
135. RightScale 2017 State of the Cloud Report, https://www.rightscale.com/2017-cloud-report, last accessed 2017/12/13.
136. IDC FutureScape: Worldwide Datacenter 2018 Predictions Description. October (2017).
137. Asyabi, E., Sharifi M. A New Approach for Dynamic Virtual Machine Consolidation in Cloud Data Centers. IJMECS, Vol.7, No.4, pp.61-66. (2015).
138. Jain, S., Sharma, V. Enhanced Load Balancing Approach to Optimize the Performance of the Cloud Service using Virtual Machine Migration. International Journal of Engineering and Manufacturing (IJEM), Vol.7, No.1, pp.41-48. (2017).
139. Kaur, A., Kaur, B., Singh D.: Optimization Techniques for Resource Provisioning and Load Balancing in Cloud Environment: A Review. International Journal of Information Engineering and Electronic Business (IJIEEB), Vol.9, No.1, pp.28-35. (2017).
140. Butler, A., Dawson, P., Palmer, J., Weiss, G. J. and Yamada, K. Magic Quadrant for Integrated Systems. (2016).
141. Matthew, M. and Eric, S. Quantifying the Business Value of Nutanix Solutions. IDC. August (2015).

142. 1 Million IOPS in 1 VM – World First for HCI with Nutanix, http://longwhiteclouds.com/2017/11/14/1-million-iops-in-1-vm-world-first-for-hci-with-nutanix/, last accessed 2017/12/13.
143. Nutanix Products Series, http://www.nutanix.com/products/hardware-platforms/, last accessed 2017/12/13.
144. Hyper-Converged Infrastructure, https://www.vmware.com/products/hyper-converged-infrastructure.html, last accessed 2017/12/13.
145. Dell EMC VxRail, https://www.dellemc.com/en-ca/converged-infrastructure/vxrail/index.htm#collapse=, last accessed 2017/12/13.
146. Scale Computing Hardware Platforms, https://www.scalecomputing.com/products/hardware-platforms/, last accessed 2017/12/13.
147. R. L. Villars and S. G. Middleton, “IDC FutureScape: Worldwide Datacenter 2016 Predictions,” 2015.
148. B. Andrew, W. George J, D. Philip, Z. Stanley, R. Errol, and A. Hiroko, “Magic Quadrant for Integrated Systems,” 2015.
149. H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, “Vision and challenges for realising the Internet of Things.” © European Union, Luxembourg: Publications Office of the European Union, p. 229, 2010.
150. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of things: A survey on enabling technologies, protocols, and applications,” Communications Surveys \& Tutorials, IEEE, vol. 17, no. 4, pp. 2347–2376, 2015.
151. "M2M Sector Map," Beecham Research, Sep. 2011. [Online] Available from: http://www.beechamresearch.com/article.aspx?id=4 [Accessed Mar 20, 2017]
152. J. Gantz and D. Reinsel, “The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east,” IDC iView: IDC Anal. Future, vol. 2007, pp. 1–16, Dec. 2012.
153. ETSI Technical Specification, “Machine-to-Machine Communications (M2M); M2M Service Requirements.” Technical Specification. ETSI TS 102 689 V1.1.1(2010-08).
154. J. Belissent, Getting Clever About Smart Cities: New Opportunities Require New Business Models, Forrester Research, 2010.
155. O. Vermesan and P. Friess, Internet of Things - From Research and Innovation to Market Deployment. River Publishers, 2014.
156. W. P. Turner IV, J. H. Seader, and V. E. Renaud, “Data center site infrastructure tier standard: Topology,” Uptime Institute, 2010.
157. A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, and others, “Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google’s Datacenter Network,” in Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, 2015, pp. 183–197.
158. Windows Containers [Online] Available from: https://msdn.microsoft.com/en-us/virtualization/windowscontainers/about/about_overview [Accessed Jan 20, 2016]

159. Docker Solutions and Use Cases [Online] Available from: https://www.docker.com/products/use-cases [Accessed Jan 20, 2016]
160. Getting Started with Nano Server [Online] Available from: https://technet.microsoft.com/en-us/library/mt126167.aspx [Accessed Jan 20, 2016]
161. S. D. Lowe, "2015 State of Hyperconverged Infrastructure Market Report," ActualTech Media, May 2015.
162. "G00266749 Magic Quadrant for Integrated Systems," Gartner, August 11, 2015.
163. 13 Powerful Hyper-Converged Infrastructure Solutions, [Online] Available from: http://www.crn.com/slide-shows/virtualization/300076666/13-powerful-hyper-converged-infrastructure-solutions.htm/pgno/0/13 [Accessed Jan 20, 2016]
164. 10 Hyperconvergence Trendsetters, [Online] Available from: http://www.networkcomputing.com/storage/10-hyperconvergence-trendsetters/1693364718 [Accessed Jan 20, 2016]
165. Breiter, G., Behrendt, M., Gupta, M., Moser, S. D., Schulze, R., Sippli, I., & Spatzier, T. (2014). Software defined environments based on TOSCA in IBM cloud implementations. IBM Journal of Research and Development, 58(2/3), 9-1.
166. Kalantar, M. H., Rosenberg, F., Doran, J., Eilam, T., Elder, M. D., Oliveira, F., ... & Roth, T. (2014). Weaver: Language and runtime for software defined environments. IBM Journal of Research and Development, 58(2/3), 10-1.
167. Arnold, W. C., Arroyo, D. J., Segmuller, W., Spreitzer, M., Steinder, M., & Tantawi, A. N. (2014). Workload orchestration and optimization for software defined environments. IBM Journal of Research and Development, 58(2/3), 11-1.
168. N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," ACM SIGCOMM Computer Communication Review, vol. 44, no. 2, pp. 87–98, 2014.
169. Liu, J., Li, Y., Chen, M., Dong, W., & Jin, D. (2015). Software-defined internet of things for smart urban sensing. Communications Magazine, IEEE, 53(9), 55-63.
170. Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645-1660.
171. Botta, A., De Donato, W., Persico, V., & Pescapé, A. (2014). On the integration of cloud computing and internet of things. The 2nd International Conference on Future Internet of Things and Cloud (FiCloud), 23-30.
172. Tao, F., Cheng, Y., Da Xu, L., Zhang, L., & Li, B. H. (2014). CCIoT-CMfg: cloud computing and internet of things-based cloud manufacturing service system. IEEE Transactions on Industrial Informatics, 10(2), 1435-1442.
173. Aazam, M., Khan, I., Alsaffar, A., & Huh, E. (2014). Cloud of Things: Integrating Internet of Things and cloud computing and the issues involved. In Proceedings of 11th International Bhurban Conference on Applied Sciences & Technology (IBCAST) Islamabad, Pakistan, 414-419.
174. S. Verma, Y. Kawamoto, Z. M. Fadlullah, H. Nishiyama and N. Kato, "A Survey on Network Methodologies for Real-Time Analytics of Massive IoT Data and Open

Research Issues," in IEEE Communications Surveys & Tutorials, vol. 19, no. 3, pp. 1457-1477, thirdquarter 2017. doi: 10.1109/COMST.2017.2694469

175. IEEE Standards Association et al. P2413-Standard for an Architectural Framework for the Internet of Things (IoT) //Institute of Electrical and Electronics Engineers, New York. – 2016.
176. X. Fafoutis, A. Elsts, R. Piechocki and I. Craddock, "Experiences and Lessons Learned From Making IoT Sensing Platforms for Large-Scale Deployments," in IEEE Access, vol. 6, pp. 3140-3148, 2018. doi: 10.1109/ACCESS.2017.2787418
177. Tomar P., Kaur G. Examining cloud computing technologies through the internet of things. – 2017. IGI Global. – Pages: 311
178. Pires, F.L., Barán, B. A virtual machine placement taxonomy. In: 15th IEEE/ACM In-ternational Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp. 159–168. (2015).
179. Calcavecchia, N., Biran, O., Hadad, E. and Moatti, Y. VM Placement Strategies for Cloud Scenarios. In 5th IEEE International Conference on Cloud Computing CLOUD, pp. 852-859. (2012).
180. Gao, Y., Guan, H., Qi, Z., Hou, Y. and Liu, L. A multi-objective ant colony system al-gorithm for virtual machine placement in cloud computing. Journal of Computer and System Sciences, vol. 79, no. 8, pp. 1230–1242 (2013).
181. Mark CC, Niyato D, Chen-Khong T. Evolutionary optimal virtual machine placement and demand forecaster for cloud computing. In: IEEE International Conference on Advanced Information Networking and Applications (AINA), pp. 348–355. (2011).
182. Wu, Y., Tang, M. and Fraser, W. A simulated annealing algorithm for energy efficient virtual machine placement. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 1245–1250. (2012).
183. Kaleem, M.A. and Khan, P.M. Commonly used simulation tools for cloud computing research. In: 2nd International Conference on Computing for Sustainable Global De-velopment (INDIACom), pp. 1104–1111. (2015).
184. Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A. and Buyya, R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, vol. 41, no. 1, pp. 23–50 (2011).
185. Salimian, L. and Safi, F. Survey of energy efficient data centers in cloud computing. In: 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, IEEE Computer Society, pp. 369–374. (2013).
186. Mills, K., Filliben, J. and Dabrowski, C. Comparing VM-placement algorithms for on-demand clouds. In: IEEE Third International Conference on Cloud Computing Tech-nology and Science (CloudCom), pp. 91–98. (2011).
187. Ferreto, T., De Rose, C. and Heiss, H.U. Maximum migration time guarantees in dynamic server consolidation for virtualized data centers. In: Euro-Par 2011 Parallel Processing, pp. 443–454. Springer (2011).

188. Shigeta, S., Yamashima, H., Doi, T., Kawai, T. and Fukui, K. Design and implementation of a multi-objective optimization mechanism for virtual machine placement in cloud computing data center. Cloud Computing, pp. 21–31. Springer (2013).
189. Cao, Z. and Dong, S. An energy-aware heuristic framework for virtual machine consolidation in cloud computing. The Journal of Supercomputing, pp. 1–23 (2014).
190. Sun, M., Gu, W., Zhang, X., Shi, H. and Zhang, W. A matrix transformation algorithm for virtual machine placement in cloud. In: 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1778–1783. (2013).
191. Pires, F.L. and Barán, B.: Multi-objective virtual machine placement with service level agreement: A memetic algorithm approach. In: 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, pp. 203–210. IEEE Computer Society (2013).
192. Wang, W., Chen, H. and Chen, X. An availability-aware virtual machine placement approach for dynamic scaling of cloud applications. In: 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC), pp. 509–516. (2012).
193. Masson, R., Vidal, T., Michallet, J., Penna, P.H.V., Petrucci, V., Subramanian, A. and Dubedout, H. An iterated local search heuristic for multi-capacity bin packing and ma-chine reassignment problems. Expert Systems with Applications, vol. 40, no. 13, pp. 5266–5275 (2013).
194. Beloglazov A. Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers/ Concurrency and Computation: Practice and Experience. – 2012. – vol. 24, no. 13. – pp. 1397–1420.
195. Li X, Qian Z, Chi R, Zhang B, Lu S. Balancing Resource Utilization for Continuous Virtual Machine Requests in Clouds. In: 6th IEEE International Conference on Innova-tive Mobile and Internet Services in Ubiquitous Computing, IMIS, pp. 266-273. (2012).
196. Limits on Simultaneous Migrations, https://docs.vmware.com/en/VMware-vSphere/6.0/com.vmware.vsphere.vcenterhost.doc/GUID-25EA5833-03B5-4EDD-A167-87578B8009B3.html, last accessed [Accessed May 12, 2018]
197. Amazon Usage Estimates, http://blog.rightscale.com/2009/10/05/amazon-usage-estimates/, last accessed 2017/07/10
198. Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. Optimization by simulated annealing. Science, vol. 220, pp. 671-680 (1983).
199. N. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E. Equation of state calculations by fast computing machines. The journal of chemical physics, vol. 21, pp. 1087-1092 (1953).
200. CloudSim: A Framework For Modeling And Simulation Of Cloud Computing Infrastructures And Services, https://github.com/Cloudslab/cloudsim, last accessed 2017/07/10

201. T. Saber, A. Ventresque, I. Brandic, J. Thorburn and L. Murphy, Towards a Multi-objective VM Reassignment for Large Decentralised Data Centres, 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), Limassol, 2015, 65-74.
202. Eucalyptus community [Online] – Available from: http://open.eucalyptus.com/
203. S. Lee, R. Panigrahy, V. Prabhakaran, V. Ramasubrahmanian, K. Talwar, L. Uyeda, and U. Wieder, Validating heuristics for virtual machines consolidation, Microsoft Research, MSR-TR-2011- 9, 2011.
204. B. Sharma, V. Chudnovsky, J. L. Hellerstein, R. Rifaat, and C. R. Das, Modeling and synthesizing task placement constraints in google compute clusters, In Proceedings of the 2nd ACM Symposium on Cloud Computing (SOCC), 2011.
205. Теленик С.Ф. Управління навантаженням і ресурсами центрів оброблення даних при виділених серверах [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов, Р.В. Римар, К. О. Ролік.– Автоматика. Автоматизація. Електротехнічні комплексі та системі. – 2009. – №2 (24). – С.122 – 136.
206. Sutton R.S. Reinforcement Learning: An Introduction [Текст]: /R.S.Sutton, A.G.Barto //MIT Press.– 1998. – c.360
207. Beloglazov A. Energy Efficient Resource Management in Virtualized Cloud Data Centers [Текст]: матеріали of 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, The University of Melbourne, Australia, 2010: тези / Anton Beloglazov, Rajkumar Buyya. – С.826 – 831.
208. Dutreilh X. From data center resource allocation to control theory and back [Текст]: /X. Dutreilh, A. Moreau, J. Malenfant, N. Rivierre, and I. Truck //Cloud Computing.– 2010. – С.410 – 417.
209. Watkins C. J. C. H., Dayan P. Technical note: Q-learning [Текст]: /C. J. C. H. Watkins and P. Dayan //Machine Learning.– 1992. – №3(8). – С.279–292.
210. Cheng-Zhong Xu. URL: A Unified Reinforcement Learning Approach for Autonomic Cloud Management [Текст]: Department of Electrical & Computer Engineering Wayne State University, Detroit, Michigan: тези / Cheng-Zhong Xu, Jia Rao, Xiangping Bu. – С.1 – 15.
211. Теленик С.Ф. Управління ресурсами центрів оброблення даних [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов, К. Крижова //Вісник Львів. УНТУ. – Серія прикл. матем. інформ., 2009. – Вип. 15. – С.325 – 340.
212. Теленик С.Ф. Генетичні алгоритми вирішення задач управління ресурсами і навантаженням центрів оброблення даних [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов, С.А. Андросов.– Автоматика. Автоматизація. Електротехнічні комплекси та системи. – 2010. – №1 (25). – С.106 – 120.
213. M. Cafaro and G. Aloisio, “Grids, clouds, and virtualization,” in Grids, Clouds and Virtualization, Springer, pp. 1–21, 2011.
214. M. Garcia-Valls, T. Cucinotta, C. Lu, “Challenges in real-time virtualization and predictable cloud computing,” Journal of Systems Architecture, vol. 60, no. 9, pp. 726–740, 2014.

215. Hameed A., Khoshkbarforoushha A., Ranjan R., Jayaraman P. P., Kolodziej J., Balaji P., Zeadally S., Malluhi Q. M., Tziritas N., Vishnu A. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems/ Computing. – 2014. – pp. 1–24.
216. Singh S., Chana I. A Survey on Resource Scheduling in Cloud Computing: Issues and Challenges/ Journal of Grid Computing. – 2016. – pp. 1–48.
217. G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation", in Proceedings of the IEEE International Conference on Autonomic Computing (ICAC), pp. 65–73, 2006.
218. F. Farahnakian, P. Liljeberg, and J. Plosila, "Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning," in Proceedings of the 22nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2014, pp. 500–507.
219. J. Rao, X. Bu, C.-Z. Xu, L. Wang, and G. Yin, "Vconf: a reinforcement learning approach to virtual machine autoconfiguration," in Proceedings of the 6th International Conference on Autonomic Computing (ICAC), pp. 137-146, 2009.
220. D. Weerasiri, M. C. Barukh, B. Benatallah, Q. Z. Sheng, and R. Ranjan, "A taxonomy and survey of cloud resource orchestration techniques," ACM Computing Surveys (CSUR), vol. 50, no. 2, p. 26, 2017.
221. A. Yousafzai, A. Gani, R. M. Noor, M. Sookhak, H. Talebian, M. Shiraz, and M. K. Khan, "Cloud resource allocation schemes: review, taxonomy, and opportunities," Knowledge and Information Systems, vol. 50, no. 2, pp. 347–381, 2017.
222. L. Chen and H. Shen, "Considering resource demand misalignments to reduce resource over-provisioning in cloud datacenters," in INFOCOM 2017-IEEE Conference on Computer Communications, IEEE, 2017, pp. 1–9.
223. H. Shen and L. Chen, "Distributed autonomous virtual resource management in datacenters using finite-markov decision process," IEEE/ACM Transactions on Networking, vol. 25, no. 6, pp. 3836–3849, 2017.
224. M. A. H. Monil and A. D. Malony, "QoS-Aware Virtual Machine Consolidation in Cloud Datacenter," in 2017 IEEE International Conference on Cloud Engineering (IC2E), 2017, pp. 81–87.
225. X. Zhou, K. Wang, W. Jia, and M. Guo, "Reinforcement learning-based adaptive resource management of differentiated services in geo-distributed data centers," in 2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS), 2017, pp. 1–6.
226. I. Sarji, C. Ghali, A. Chehab, and A. Kayssi, "Cloudese: Energy efficiency model for cloud computing environments," in 2011 International Conference on Energy Aware Computing (ICEAC), 2011, pp. 1–6.
227. S. L. Xi, M. Guevara, J. Nelson, P. Pensabene, and B. C. Lee, "Understanding the critical path in power state transition latencies," in 2013 IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2013, pp. 317–322.

228. A. Gandhi, M. Harchol-Balter, and M. A. Kozuch, “Are sleep states effective in data centers?,” in 2012 International Green Computing Conference (IGCC), 2012, pp. 1–10.
229. A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M.Kozuch. “AutoScale: dynamic, robust capacity management for multi-tier data centers,” ACM Trans. on Computer Systems, vol. 30(4), pp. 1–26, 2012.
230. A. Paya and D. C. Marinescu, “Energy-aware load balancing and application scaling for the cloud ecosystem,” IEEE Transactions on Cloud Computing, vol. 5, no. 1, pp. 15–27, 2017.
231. J. Yang, S. Zhang, X. Wu, Y. Ran, and H. Xi, “Online Learning-Based Server Provisioning for Electricity Cost Reduction in Data Center,” IEEE Transactions on Control Systems Technology, vol. 25, no. 3, pp. 1044–1051, 2017.
232. Standard Performance Evaluation Corporation [Online] Available from: http://spec.org/ [Accessed May 12, 2018].
233. Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari, “Server workload analysis for power minimization using consolidation,” In Proceedings of USENIX ATC 2009, pp. 355–368.
234. M. R. V. Kumar and S. Raghunathan, “Power management using dynamic power state transitions and dynamic voltage frequency scaling controls in virtualized server clusters,” Turkish Journal of Electrical Engineering \& Computer Sciences, vol. 24, no. 4, pp. 2290–2306, 2016.
235. White Paper: ArchiMate® 2.0 – Understanding the Basics (W130), published by The Open Group, February 2013
236. Sjaak Laan. 2013. IT Infrastructure Architecture - Infrastructure Building Blocks and Concepts (2nd ed.). Lulu.com.
237. Gerben Wierda. Modelling networking using the networking concepts of #ArchiMate [Электронный ресурс]. — Режим доступа: http://masteringarchimate.com/2014/11/02/modelling-networking-using-the-networking-concepts-of-archimate/#more-31342 (дата обращения: 21.12.2014)
238. Александр Барсков Сеть ЦОД на проверку // «Журнал сетевых решений/LAN». 2012. № 11. [Электронный ресурс]. — Режим доступа: http://www.osp.ru/lan/2012/11/13032380/ (дата обращения: 08.04.2015).
239. Hariri S., Khargharia B., Chen H., Yang J., Zhang Y., Parashar M., Liu H. The autonomic computing paradigm/ Cluster Comput. – 2006. – vol. 9 (1). –¬ pp. 5–17.
240. Clark C., Fraser K., Hand S., Hansen J. G., Jul E., Limpach C., Pratt I., Warfield A. Live migration of virtual machines/ Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation. – 2005. – vol. 2. – pp. 273–286.
241. Wood T., Shenoy P. J., Venkataramani A., Yousif M. S. Black-box and Gray-box Strategies for Virtual Machine Migration/ NSDI. – 2007. – vol. 7. – pp. 17–17.
242. Bobroff N., Kochut A., Beaty K. Dynamic placement of virtual machines for managing SLA violations/ Integrated Network Management, 2007. IM’07. 10th IFIP/IEEE International Symposium. – 2007. – pp. 119–128.

243. Zhu X., Young D., Watson B. J., Wang Z., Rolia J., Singhal S., McKee B., Hyser C., Gmach D., Gardner R. 1000 islands: Integrated capacity and workload management for the next generation data center/ Autonomic Computing (ICAC'08). International Conference. – 2008. – pp. 172–181.
244. Shen Z., Subbiah S., Gu X., Wilkes J. Cloudscale: elastic resource scaling for multi-tenant cloud systems/ Proceedings of the 2nd ACM Symposium on Cloud Computing. – 2011. – p. 5.
245. A. Horri, M. S. Mozafari, and G. Dastghaibyfard, "Novel resource allocation algorithms to performance and energy efficiency in cloud computing," The Journal of Supercom-puting, vol. 69, no. 3, pp. 1445–1461, 2014.
246. Venticinque, Salvatore, Luca Tasquier, and Beniamino Di Martino. "Agents based cloud computing interface for resource provisioning and management," In Proceedings of the Sixth International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), pp. 249-256. IEEE, 2012.
247. Al-Ayyoub M., Jararweh Y., Daraghmeh M., Althebyan Q. Multi-agent based dynamic resource provisioning and monitoring for cloud computing systems infrastructure/ Cluster Computing. – 2015. – vol. 18, no. 2. – pp. 919–932.
248. A. Darabseh, M. Al-Ayyoub, Y. Jararweh, E. Benkhelifa, M. Vouk, and A. Rindos, "SDDC: A Software Defined Datacenter Experimental Framework," In Proceedings of the FICLOUD 2015, 3rd International Conference on Future Internet of Things and Cloud, 2015, pp. 189–194.
249. Ролик А.И. Управление уровнем услуг корпоративной ИТ-инфраструктуры на основе координатора / А.И. Ролик //Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+. – 2013. – № 59. – С. 98–105.
250. Теленик С.Ф. Адаптивный генетический алгоритм для решения класса задач распределения ресурсов ЦОД / С.Ф. Теленик, А.И. Ролик, П.С. Савченко // Вісник НТУУ «КПІ»: Інформатика, управління та обчислювальна техніка. – К.: «ВЕК+», 2011. – № 54. – С. 164–174.
251. Ролик А.И. Концепция управления корпоративной ИТ-инфраструктурой / А.И. Ролик // Вісник НТУУ «КПІ»: Інформатика, управління та обчислювальна техніка. – К.: «ВЕК+», 2012. – № 56. – С. 31–55.
252. Apache.org. HDFS Architecture. http://hadoop.apache.org/core/docs/current/hdfs_design.html
253. M. McCauley, "POX," 2012. [Online]. Available: http://www.noxrepo. org/
254. N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: towards an operating system for networks," Comp. Comm. Rev., 2008.
255. U. Krishnaswamy, P. Berde, J. Hart, M. Kobayashi, P. Radoslavov, T. Lindberg, R. Sverdlov, S. Zhang, W. Snow, and G. Parulkar, "ONOS: An open source distributed SDN OS," 2013. [Online]. Available: http://www.slideshare.net/umeshkrishnaswamy/open-network-operating-system
256. OpenDaylight, "OpenDaylight: A Linux Foundation Collaborative Project," 2013. [Online]. Available: http://www.opendaylight.org

257. D. Kreutz, F. M. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” In Proceedings of the IEEE, vol. 103, no. 1, pp. 14–76, 2015.
258. XenSource, Inc. [2008]. “The Xen™ virtual machine monitor”, http://www.cl.cam.ac.uk/research/srg/netos/projects/archive/xen/
259. VMware, Inc. [2016]. “vSphere Hypervisor,” http://www.vmware.com/products/vsphere-hypervisor/
260. Microsoft Corporation. [2016] “Hyper-V overview,” https://technet.microsoft.com/library/hh831531.aspx
261. Теленик С.Ф. Управляемый генетический алгоритм в задачах распределения виртуальных машин в ЦОД / С.Ф. Теленик, А.И. Ролик, П.С. Савченко, М.Е. Боданюк // Вісник ЧДТУ. — 2011. — № 2. — С. 104—113.
262. Pultz, J. E. (2011). DCIM, New Tools to Monitor, Manage and Control Power. Gartner Data Center Conference.
263. D. J. Cappuccio, "DCIM: Going Beyond IT" Gartner ID G00174769, May29, 2010.
264. J. Pultz, F. De Silva, and A. Adams, “Market Trends: Addressable DCIM Market,” G00239150, November 27, 2012.
265. A. M. Sampaio, J. G. Barbosa, and R. Prodan, “PIASA: A power and interference aware resource management strategy for heterogeneous workloads in cloud data centers,” Simulation Modelling Practice and Theory, vol. 57, pp. 142–160, 2015.
266. M. M. Hassan, M. S. Hossain, A. J. Sarkar, and E.-N. Huh, “Cooperative game-based distributed resource allocation in horizontal dynamic cloud federation platform,” Information Systems Frontiers, vol. 16, no. 4, pp. 523–542, 2014.
267. Зубов Д. А., Ульшин В. А., Жариков Э. В., Григоренко М. С. Концепция програмно-аппаратного комплекса долгосрочного прогнозирования средней температуры воздуха на базе статистико-гидродинамических моделей // Збірник наукових праць Українського державного геологорозвідувального інституту. 2007. №4. – С. 223-226.
268. Э. В. Жариков, В. Л. Овчинников Система передачи мультимедийных данных в локальных сетях // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2008. №9. Частина 1. – С. 142-146.
269. Zharikov E. Topical questions of implementation of information services in a network of university // TEKA Kom. Mot. i Energ. Roln. 2010. Vol 10B – С. 331-337
270. Жаріков Е.В., Солдатенко В.Ю. Методика порівняльного аналізу бездротових технологій // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2011. № 11. Частина 2. – С. 250-253.
271. Жариков Э. В. Основные направления оптимизации ит-инфраструктуры учебных заведений // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2011. № 3. – С. 66-71.
272. Zharikov E. Analysis of the theoretical and applied aspects of modern IT infrastructure // TEKA. COMMISSION OF MOTORIZATION AND ENERGETICS IN AGRICULTURE – 2013, Vol. 13, No.4, 297-306

273. Samozdra M., Zharikov E., Samozdra O. Implementation of automated informational interactions as a part of integrated information-processing system // TEKA. COMMISSION OF MOTORIZATION AND ENERGETICS IN AGRICULTURE – 2014, Vol. 14, No.1, 229-237
274. Rolik O., Telenyk S., Zharikov E. Service quality management in microcloud-based IoT infrastructure //Czasopismo Techniczne. – 2016. – vol. 2-E(12). – С. 231-244. DOI: 14467/2353737XCT.16.243.6042.
275. Теленик С.Ф., Ролик А.И., Жариков Э.В. Управление распределением виртуальных машин в ЦОД / Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. — К.: Век+. — 2016. — № 64. — С. 90—99.
276. O. Rolik, S. Telenyk, E. Zharikov, “IoT and Cloud Computing: The Architecture of Microcloud-Based IoT Infrastructure Management System,” in Emerging Trends and Applications of the Internet of Things. Hershey, PA: IGI Global, 2017, pp.198-234. doi:10.4018/978-1-5225-2437-3
277. Telenyk S., Zharikov E., Rolik O. Consolidation of Virtual Machines Using Stochastic Local Search // Advances in Intelligent Systems and Computing. – Springer, Cham, 2017. – Vol. 689. – С. 523-537. https://doi.org/10.1007/978-3-319-70581-1_37
278. Жаріков Е.В., Сердюк Е.А. Метод консолидации виртуальных машин на основе лучевого поиска / Е.В.Жаріков // Автомобіль і електроніка. Сучасні технології. – 2017. – No12. – С. 180–186.
279. Ролік О.І., Теленик С.Ф., Жаріков Е.В. Управління рівнем послуг в системі інтернету речей з мікрохмарною архітектурою / Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. — К.: Век+. — 2017. — № 65. — С. 110—117.
280. Жаріков Е.В. Керування ресурсами хмарних ЦОД на основі евристичного пошуку / Проблеми програмування. – 2017. – № 4. – С. 16-27
281. Жаріков Е. В. Динамічне розміщення віртуальних машин на основі навчання з підкріпленням в хмарних центрах обробки даних / Е. В. Жаріков, А. А. Коваль, Р. А. Терентьєв. // Наукові вісті Далівського університету. - 2017. - № 13. - Режим доступу: http://nbuv.gov.ua/UJRN/Nvdu_2017_13_4
282. Rolik O., Telenyk S., Zharikov E. Management of services of a hyperconverged infrastructure using the coordinator // Advances in Intelligent Systems and Computing. – Springer, Cham, 2018. – Vol. 754. – С. 456-467. https://doi.org/10.1007/978-3-319-91008-6_46
283. Жаріков Е. В., Моделювання динаміки хмарного центру обробки даних у просторі станів // Моделювання та інформаційні технології. – 2018. – № 84(6). – С. 125-134
284. Жаріков Е.В. Інтегроване управління ресурсами хмарного центру обробки даних на основі віртуальних машин, Математичні машини і системи, 2018, № 2 с. 21-32

285. Telenyk S., Rolik O., Zharikov O., Serdiuk Y. Energy efficient data center resources management using beam search algorithm //Czasopismo Techniczne. – 2018. – Т. 2018. – №. 4. – С. 127-138.
286. Жаріков Е. В., Структурна оптимізація моделей прогнозу споживання обчислювальних ресурсів в умовах віртуалізації // Електронне моделювання, 2018, №5, с. 49-66
287. Жаріков Е. В. Метод управління дворівневим сховищем віртуалізованого центру обробки даних / Проблеми програмування. – 2018. – № 4. – С. 3-14 https://doi.org/10.15407/pp2018.04.003
288. Жаріков Е. В. Технології віртуалізації на базі VIRTUAL SERVER в навчальному процесі // Збірник наукових праць Східноукраїнського національного університету імені Володимира Даля, Міжнародні Далівські читання XV Нуково-практична конференція "Університет і регіон:проблеми сучасної освіти"/ За заг.ред.проф. Голубенка О.Л.– Луганськ: вид-во Східноукраїнського національного університету імені Володимира Даля, 2009. – С. 247-249.
289. Жариков Э. В. Актуальные вопросы оптимизации IT-инфраструктуры учебных заведений // Комп'ютерні науки для інформаційного суспільства: Матеріали міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (вебінар) (м. Луганськ, 22-23 грудня 2010 р.). – Луганськ: Вид-во «Ноулідж», 2010. – С. 98-102.
290. Солдатенко В. Ю., Жариков Э. В. Методика сравнительного анализа беспроводных технологий для проектирования сетей передачи данных // Комп'ютерні науки для інформаційного суспільства: Матеріали міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (вебінар) (м. Луганськ, 22-23 грудня 2010 р.). – Луганськ: Вид-во «Ноулідж», 2010. – С. 126-128.
291. Кордубайло С. А., Жариков Э. В. Планирование внедрения технологий виртуализации на предприятии // Комп'ютерні науки для інформаційного суспільства: Матеріали II Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 23-24 листопада 2011 р.). – Луганськ: Вид-во «Ноулідж», 2011. – С. 109-112.
292. Альсаясні М., Жаріков Е. В. Підвищення якості роботи інформаційних сервісів корпоративної мережі // Комп'ютерні науки для інформаційного суспільства: Матеріали II Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 23-24 листопада 2011 р.). – Луганськ: Вид-во «Ноулідж», 2011. – С. 82-85.
293. Грищенко Е. В., Жариков Э. В. Уровневое разбиение элементов ИТ-инфраструктуры // Комп'ютерні науки для інформаційного суспільства: Матеріали III Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 12-13 грудня 2012 р.). – Луганськ: Вид-во «Ноулідж», 2012. – С. 107-108.

294. Жариков Э. В., Альсаясни М., Лукутин О. В. Анализ систем мониторинга ИТ-инфраструктуры и разработка подхода к их интеграции // Комп'ютерні науки для інформаційного суспільства: Матеріали III Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 12-13 грудня 2012 р.). – Луганськ: Вид-во «Ноулідж», 2012. – С. 128-132.
295. Удовенко Д. В., Жариков Э. В. Облачные вычисления как инновационное направление информационно-коммуникационных технологий // Комп'ютерні науки для інформаційного суспільства: Матеріали III Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 12-13 грудня 2012 р.). – Луганськ: Вид-во «Ноулідж», 2012. – С. 208-210.
296. Э. В. Жариков, А. Салем Повышение качества информации в современной ИТ-инфраструктуре // Информатика и вычислительная техника: сборник научных трудов 5-й Всероссийской научно-технической конференции аспирантов, студентов и молодых ученых ИВТ-2013 / под ред. Н. Н. Войта. – Ульяновск: УлГТУ, 2013. –С. 69-74
297. Э. В. Жариков, Альсаясни М. Об одном подходе к интеграции систем мониторинга в современной ИТ-инфраструктуре // Информатика и вычислительная техника: сборник научных трудов 5-й Всероссийской научно-технической конференции аспирантов, студентов и молодых ученых ИВТ-2013 / под ред. Н. Н. Войта. –Ульяновск: УлГТУ, 2013. –С. 65-69
298. Жариков Э. В. Критерии оптимизации ИТ-инфраструктуры предприятия // Информационно-компьютерные технологии в экономике, образовании и социальной сфере. Выпуск 8. – Симферополь : ФЛП Бондаренко О.А., 2013. С. 6-7
299. J. Obinna, Zharikov E. Architectural Development of Pivate Cloud for Mid Business Enterprises // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 42-46.
300. Adetola R., Zharikov E. Challenges in cloud computing // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 33-35
301. Жариков Э. В., Алдееб С. Неструктурированная информация предприятия, качественный и количественный аспект // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 52-56

302. Жариков Э. В., Альсаясни М., Лукутин О. В. Идентификация инцидента методом распознавания образов в системах мониторинга // Комп'ютерні науки для інформаційного суспільства: Матеріали IV Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Луганськ, 11-12 грудня 2013 р.). – Луганськ: Вид-во «Ноулідж», 2013. – С. 56-58
303. Жариков Э. В., Алдееб С. Моделирование ИТ-инфраструктуры с использованием Archimate // Комп'ютерні науки для інформаційного суспільства: Матеріали V Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених (веб-конференція) (м. Сєвєродонецьк, 23 грудня 2014 р.). – Сєвєродонецьк: Вид-во СНУ ім.В.Даля, 2014. – С. 21-26
304. Жариков Э. В., Алдееб С. Анализ компонентов ИТ-инфраструктуры в контексте обработки информации предприятия // Комп'ютерні інтелектуальні системи та мережі: Матеріали VIII Всеукраїнської науково практичної WEB конференції аспірантів, студентів та молодих вчених (24-26 березня 2015 р.). – Кривий Ріг: ДВНЗ «Криворізький національний університет», 2015. – С. 23-26
305. Жариков Э. В. Архитектура системы управления и мониторинга производительности программно-определяемой ИТ-инфраструктуры // «АВІА-2015»: Матеріали XII Міжнародної науково-технічної конференції (28-29 квітня 2015 р.). – Київ: Національний авіаційний університет, 2015. – С. 638-641
306. Жариков Э. В. Гиперконвергентная архитектура, анализ возможностей применения в ЦОД // Комп'ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції аспірантів, студентів та молодих вчених (22-24 березня 2016 р.). – Кривий Ріг: ДВНЗ «Криворізький національний університет», 2016. – С. 14-19
307. Telenyk S. An approach to Software Defined Cloud Infrastructure management / S. Telenyk, E. Zharikov, O. Rolik // Proc. of the XI International Scientific and Technical Conference "Computer Science and Information Technologies Congress on Information Technology" (CSIT 2016) 6–10 September, Lviv, Ukraine. – 2016 p. – p. 21–26. http://dx.doi.org/10.1109/STC-CSIT.2016.7589859
308. S. Telenyk, E. Zharikov, O. Rolik, "An approach to virtual machine placement in cloud data centers," In proc. of the 2016 International Conference Radio Electronics & Info Communications (UkrMiCo) 11–16 September, Kyiv, Ukraine. – 2016 p. – p. 1–6. http://dx.doi.org/10.1109/UkrMiCo.2016.7739645
309. Rolik O., Zharikov E., Telenyk S. Microcloud-based architecture of management system for IoT infrastructures //Problems of Infocommunications Science and Technology (PIC S&T), 2016 Third International Scientific-Practical Conference. – IEEE, 2016. – pp. 149-151. DOI: 10.1109/INFOCOMMST.2016.7905363
310. Rolik, O., Telenyk, S., Zharikov, E., & Yasochka, M. (2016, December). Decomposition-compensation approach to microcloud-based IoT infrastructure management. In Proc. of 3rd World Forum on Internet of Things (WF-IoT) 2016 (pp. 603-608). IEEE. DOI: 10.1109/WF-IoT.2016.7845413

311. Rolik O., Telenyk S., Zharikov E., Samotyy V., Dynamic Virtual Machine Allocation Based on Adaptive Genetic Algorithm // The Eighth International Conference on Cloud Computing, GRIDs, and Virtualization. – 2017. – pp. 108-114.
312. Коваль А., Жаріков Е. Порівняльний аналіз існуючих методів управління ресурсами в умовах хмарних обчислень // Комп'ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції КІСМ-2017 (22-24 березня 2017 р.). – Кривий Ріг: ДВНЗ «Криворізький національний університет», 2017. – С. 8-10
313. Сягайло Т., Жаріков Е. Порівняльний аналіз алгоритмів для управління розміщенням віртуальних машин // Комп'ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції КІСМ-2017 (22-24 березня 2017 р.). – Кривий Ріг: ДВНЗ «Криворізький національний університет», 2017. – С. 10-13
314. Терентьєв Р., Жаріков Е. Порівняльний аналіз засобів моделювання інфраструктури хмарних обчислень // Комп'ютерні інтелектуальні системи та мережі: Матеріали IX Всеукраїнської науково практичної WEB конференції КІСМ-2017 (22-24 березня 2017 р.). – Кривий Ріг: ДВНЗ «Криворізький національний університет», 2017. – С. 13-16
315. Andrii Koval, Roman Terentiev, Eduard Zharikov, Comparative Analysis of Modeling Methods of Infrastructure of Cloud Computing, Science and Technology of the XXI Century: the XVIII All-Ukrainian Students R&D Conference Proceeding, (Kyiv, December 07, 2017) / National Technical University of Ukraine „Igor Sikorsky Kyiv Polytechnic Institute". – Part IV. – Kyiv, 2017. – P.124
316. R. Terentiev, A. Koval, E. Zharikov, Comparative Analysis of Resource Management Methods in Cloud Computing, Science and Technology of the XXI Century: the XVIII All-Ukrainian Students R&D Conference Proceeding, (Kyiv, December 07, 2017) / National Technical University of Ukraine „Igor Sikorsky Kyiv Polytechnic Institute". – Part V. – Kyiv, 2017. – P.125
317. Сягайло Т.А. Способи налаштування сховищ в гіперконвергентних системах./ Жаріков Е.В., Сягайло Т.А., Коваль А.А. // Актуальні наукові дослідження в сучасному світі. - 2018. Вып. 4(36), ч. 3 - С.41 - 51.
318. Жаріков Е.В., Сердюк Є.О. Консолідація віртуальних машин з використанням променевого пошуку // Інформатика та обчислювальна техніка ІОТ-2017. – Київ: НТУУ "КПІ ім. І. Сікорського", 2017. – С. 79-84
319. Rolik O., Zharikov E., Kolesnik V., Yasochka M., "Rule-based Algorithmic Approach for Solving Problems of Impact Analysis in Access Networks," In proc. of the 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON) May 29 – June 2, Kyiv, Ukraine. – 2017 – p. 1161–1166. DOI: 10.1109/UKRCON.2017.8100432
320. Telenyk S., Zharikov E., and Rolik O. "Consolidation of Virtual Machines Using Simulated Annealing Algorithm" in Proc. of the XIIth International Scientific and Technical Conference "Computer Science and Information Technologies" (CSIT

2017) 5–8 September, Lviv, Ukraine. – 2017. – pp. 117–121. DOI: 10.1109/STC-CSIT.2017.8098750

321. Telenyk S., Bidyuk P., Zharikov E. and Yasochka M., "Assessment of cloud service provider quality metrics," 2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo), Odessa, 2017, pp. 1-5. doi: 10.1109/UkrMiCo.2017.8095422"
322. Telenyk S., Zharikov E., Rolik O., "An Integrated Approach to Cloud Data Center Resource Management" //Problems of Infocommunications Science and Technology (PIC S&T), 4th International Scientific-Practical Conference. – IEEE, 2017. – pp. 211-218. DOI: 10.1109/INFOCOMMST.2017.8246382
323. Telenyk S., Rolik O., Zharikov O., Serdiuk Y. "Consolidating Virtual Machines with the Use of Beam Search Algorithm", The Fourth International Conference on Automatic Control and Information Technology (ICACIT'17), 2017, pp. 34-46
324. O. Rolik, E. Zharikov, A. Koval and S. Telenyk, "Dynamic management of data center resources using reinforcement learning," 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 2018, pp. 237-244. doi: 10.1109/TCSET.2018.8336194"
325. Коваль А.А., Жаріков Е.В. Метод розміщення віртуальних машин на основі навчання з підкріпленням // Інформатика та обчислювальна техніка: Матеріали наукової конференції студентів, магістрантів та аспірантів ІОТ-2018 (23 – 24 квітня 2018). – Київ: НТУУ "КПІ ім. І. Сікорського", 2018. – С. 29-35.
326. Сягайло Т.А., Жаріков Е.В., Управління процесом збереження даних в гіперконвергентних системах // Інформатика та обчислювальна техніка: Матеріали наукової конференції студентів, магістрантів та аспірантів ІОТ-2018 (23 – 24 квітня 2018). – Київ: НТУУ "КПІ ім. І. Сікорського", 2018. – С. 119-122.
327. Терентьєв Р. А., Жаріков Е.В., Прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень // Інформатика та обчислювальна техніка: Матеріали наукової конференції студентів, магістрантів та аспірантів ІОТ-2018 (23 – 24 квітня 2018). – Київ: НТУУ "КПІ ім. І. Сікорського", 2018. – С. 123-126.
328. Zharikov E., Rolik O., Telenyk S. A Decomposition Approach to Hierarchical Management of Cloud Data Center Services //2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). – IEEE, 2018. – Т. 1. – С. 42-47.
329. Rolik O., Zharikov E., Yasochka M., Butenko M. The Method of Impact Analysis for Access Networks with RIP and OSPF Protocols // 2018 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo), Odesa, Ukraine, 2018, pp. 1-7.
330. S. Telenyk, E. Zharikov and O. Rolik, "Modeling of the Data Center Resource Management Using Reinforcement Learning," 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2018, pp. 289-296. doi: 10.1109/INFOCOMMST.2018.8632064

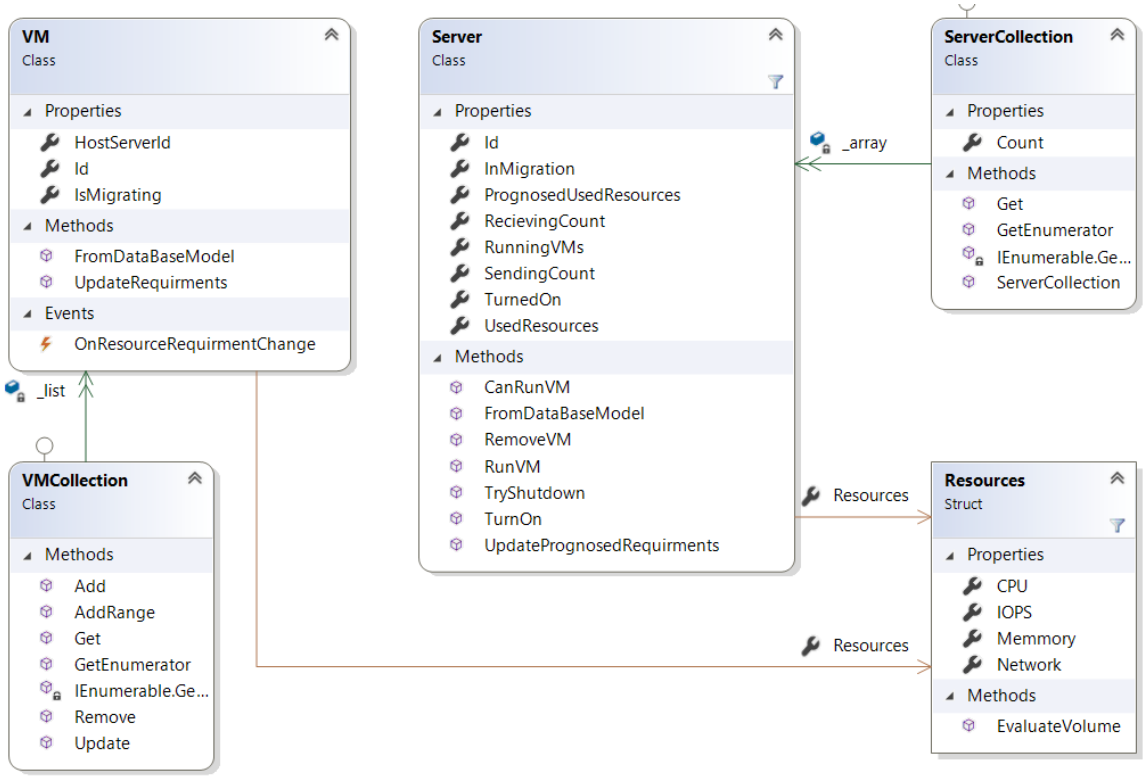
331. Camacho E. F., Alba C. B. Model predictive control. – Springer Science & Business Media, 2013.
332. Кириллов И. Облака и ЦОД - что принес нам этот год? / Игорь Кириллов. // СиБ. – 2018. – №6. – С. 18–28.
333. Кириллов И. Все выше и выше. Рынок облачных услуг в Украине стремительно растет. / Игорь Кириллов. // СиБ. – 2018. – №3(100). – С. 36–46.
334. Кириллов И. На буксире облаков. Украинский рынок ЦОД продолжает развитие. / Игорь Кириллов. // СиБ. – 2018. – №3(100). – С. 18–28.
335. Cisco Global Cloud Index: Forecast and Methodology, 2016–2021 White Paper, November 19, 2018,
336. Jim Handy, Thomas Coughlin, Survey Update: Users Share Their 2017 Storage Performance Needs, white paper, Storage networking industry association, 2017, 7 С.
337. Rightscale 2019 state of the cloud report from Flexera, white paper, Flexera, 2019, С. 50.
338. Морозов В. К., Долганов А. В. Основы теории информационных сетей: Учеб. пособие. -М.: Высш.шк., 1987.-271с.:ил.
339. Куликовский Л. Ф., Морозов В. К. Основы информационной техники. М., 1977.
340. Куликовский Л. Ф., Мотов В. В., Теоретические основы информационных процессов: Учебное пособие для вузов по спец. "Автоматизация и механизация процессов обработки и выдачи информации". -М.: Высш. шк. ,1987. - 248 с.
341. Service Oriented Architecture: What Is SOA? http://www.opengroup.org/soa/source-book/soa/soa.htm#soa_definition
342. White Paper: Understanding the basics – an introduction to the Archimate® modeling language, version 3.0.1 (W175), published by The Open Group, November 2017
343. Жариков Э.В., Шмитько А.А. Автоматизация подготовки научных изданий к печати // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2006. №1(95). – С. 237-240.
344. Жариков Э.В. Публикация и использование знаний в глобальной сети // Вісник Східноукраїнського національного університету ім. В. Даля. – 2003. – №4(62). – С. 36-40.
345. Жариков Э.В. Разработка методов представления и организации знаний в распределенной экспертной системе. // Вісник Східноукраїнського національного університету ім. В. Даля. – 2005. – №3(85). – С. 89-95.
346. Zharikov E., Telenyk S., Rolik O. Method of Distributed Two-Level Storage System Management in a Data Center // Advances in Intelligent Systems and Computing. – Springer, Cham, 2019. vol 938. – С. 301-315.
347. White Paper: HP IT Service Management (ITSM) Reference Model, published by HP, Jun 2003
348. Telenyk S., Nowakowski G., Zharikov E., Vovk Y. Information technology for web-applications design and implementation // Адаптивні системи автоматичного управління, К: Політехніка. – 2019. – Т.1, №34. – С. 138-151.

349. Telenyk S., Zharikov E. Operator form to formulate, analyze and solve the cloud data center IT infrastructure management tasks // Адаптивні системи автоматичного управління, К: Політехніка. – 2019. – Т.2, №35. – С. 25-39.
350. da Rosa Righi, R., Lehmann, M., Gomes, M. M., Nobre, J. C., da Costa, C. A., Rigo, S. J., ... & de Oliveira, L. R. B. (2019). A Survey on Global Management View: Toward Combining System Monitoring, Resource Management, and Load Prediction. Journal of Grid Computing, 1-30.
351. Управление корпоративной ИТ-инфраструктурой [Текст] / А. И. Ролик, С. Ф. Теленик, М. В. Ясочка. – Киев: Наукова думка, 2018. – 576 с.
352. Бутстреп- аналіз якості розв'язання задач ідентифікації [Текст]: автореф. дис. д-ра технічний наук: 05.13.03 / Архипов Олександр Євгенійович ; Національний технічний ун-т України "Київський політехнічний ін-т". - К., 1995. - 33 с.
353. О. Архипов, С. Архипова, Адаптивний підхід до обробки даних експертного оцінювання при вирішенні завдань у сфері захисту інформації // Захист інформації. – 2019. – №3(21). – С. 158-167.

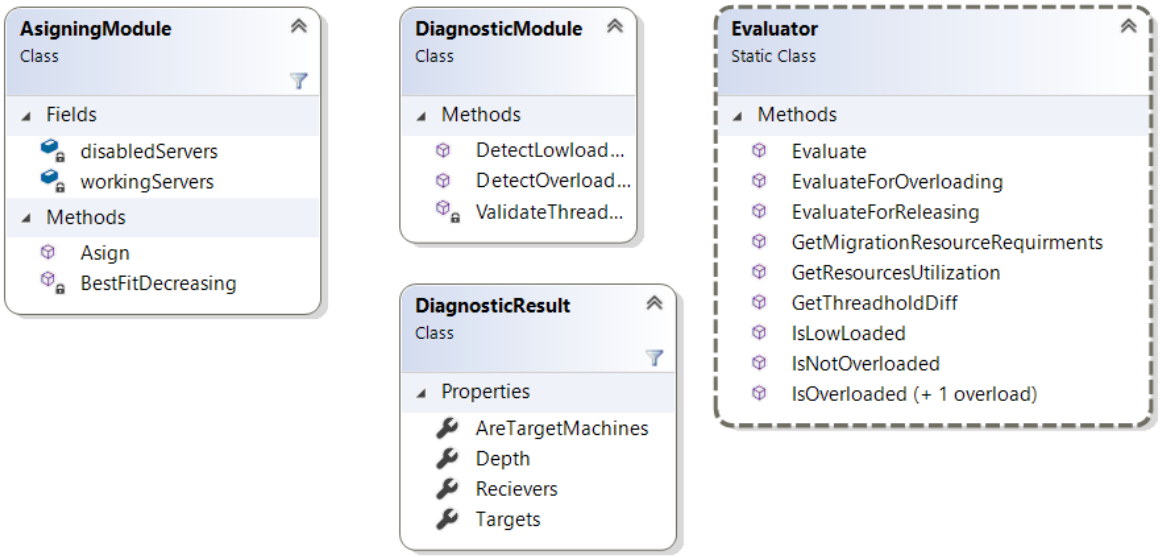
ДОДАТОК А

Діаграми класів для моделей та окремих компонентів двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку в схемі реалізації стратегії S_2.

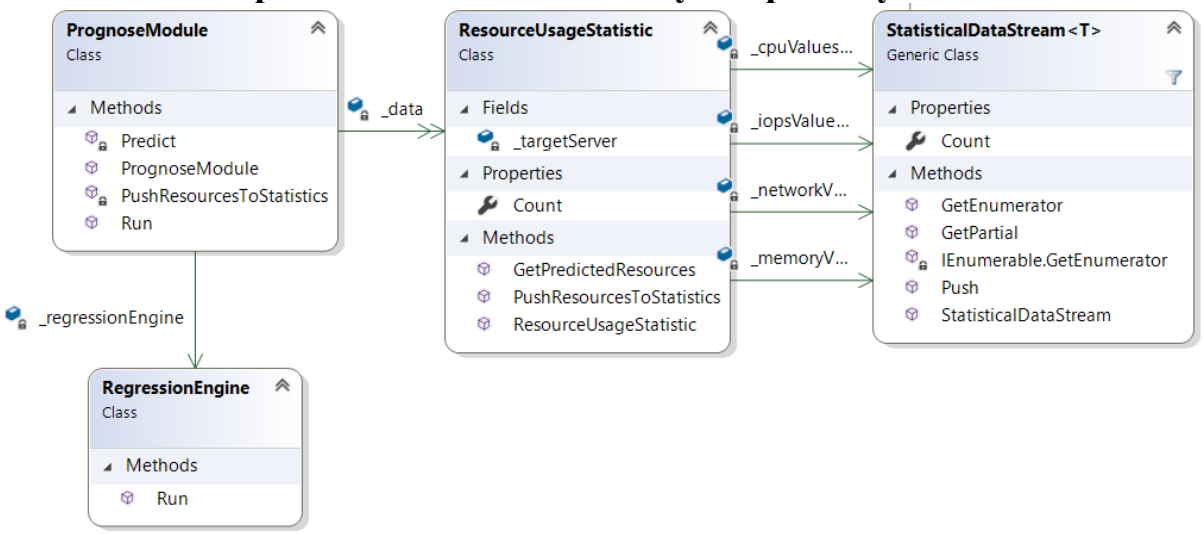
Представлення класів моделей



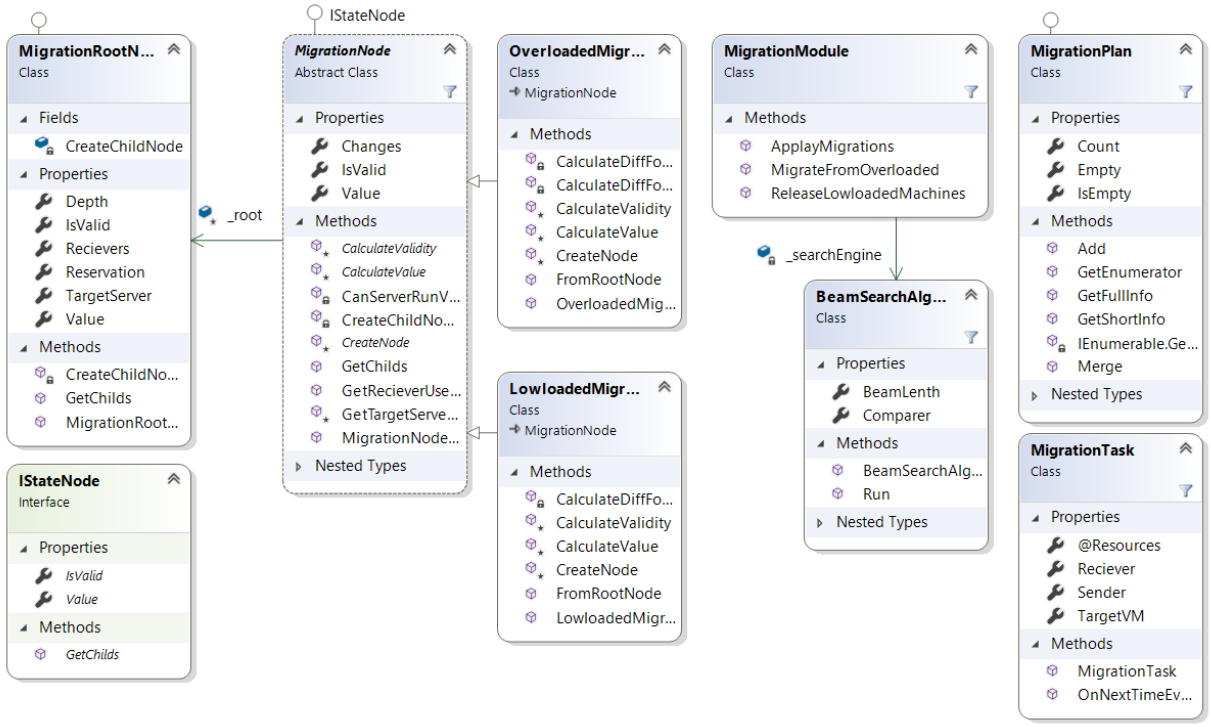
Представлення класів модулів діагностики та призначення



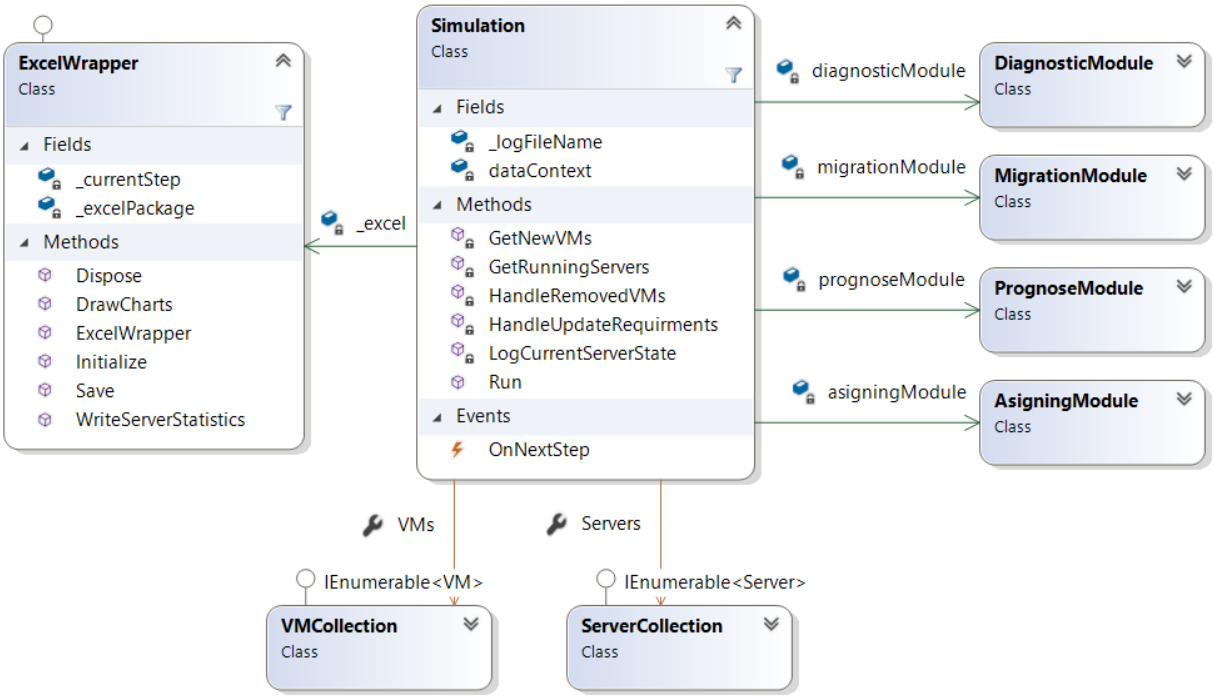
Представлення класів модуля прогнозування



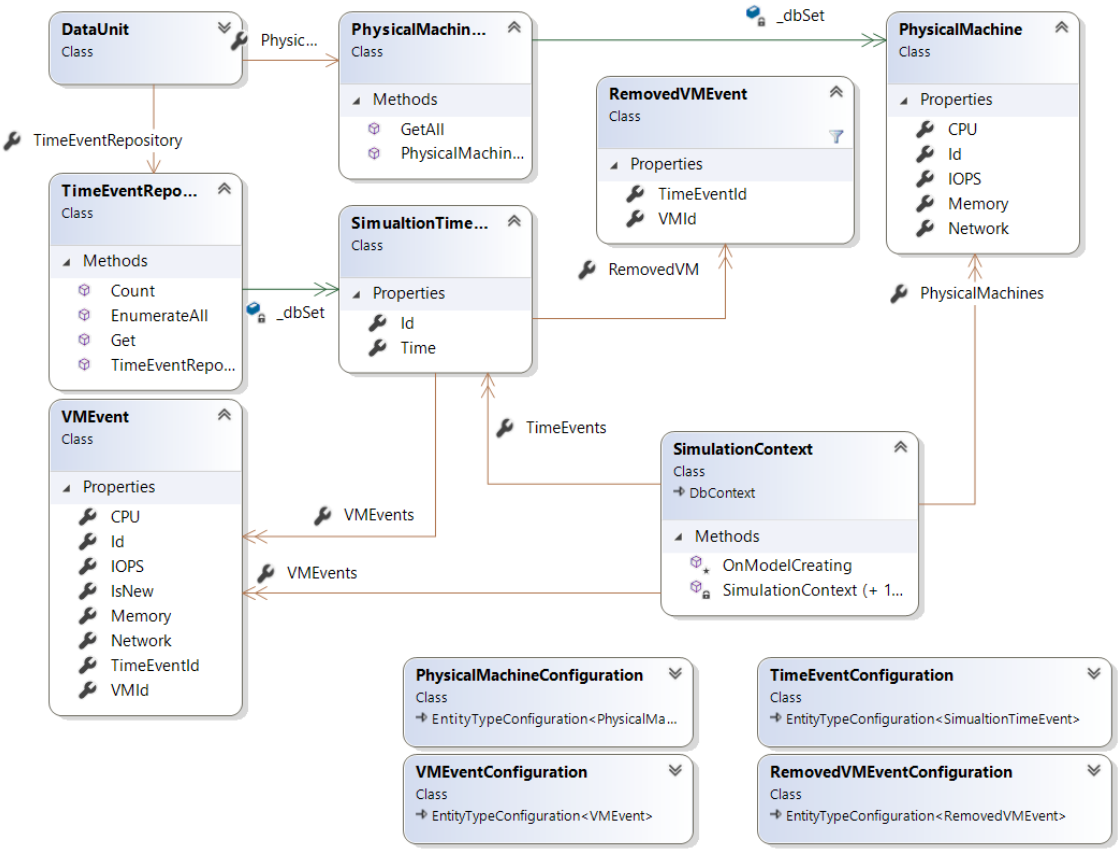
Представлення класів модуля міграцій ВМ



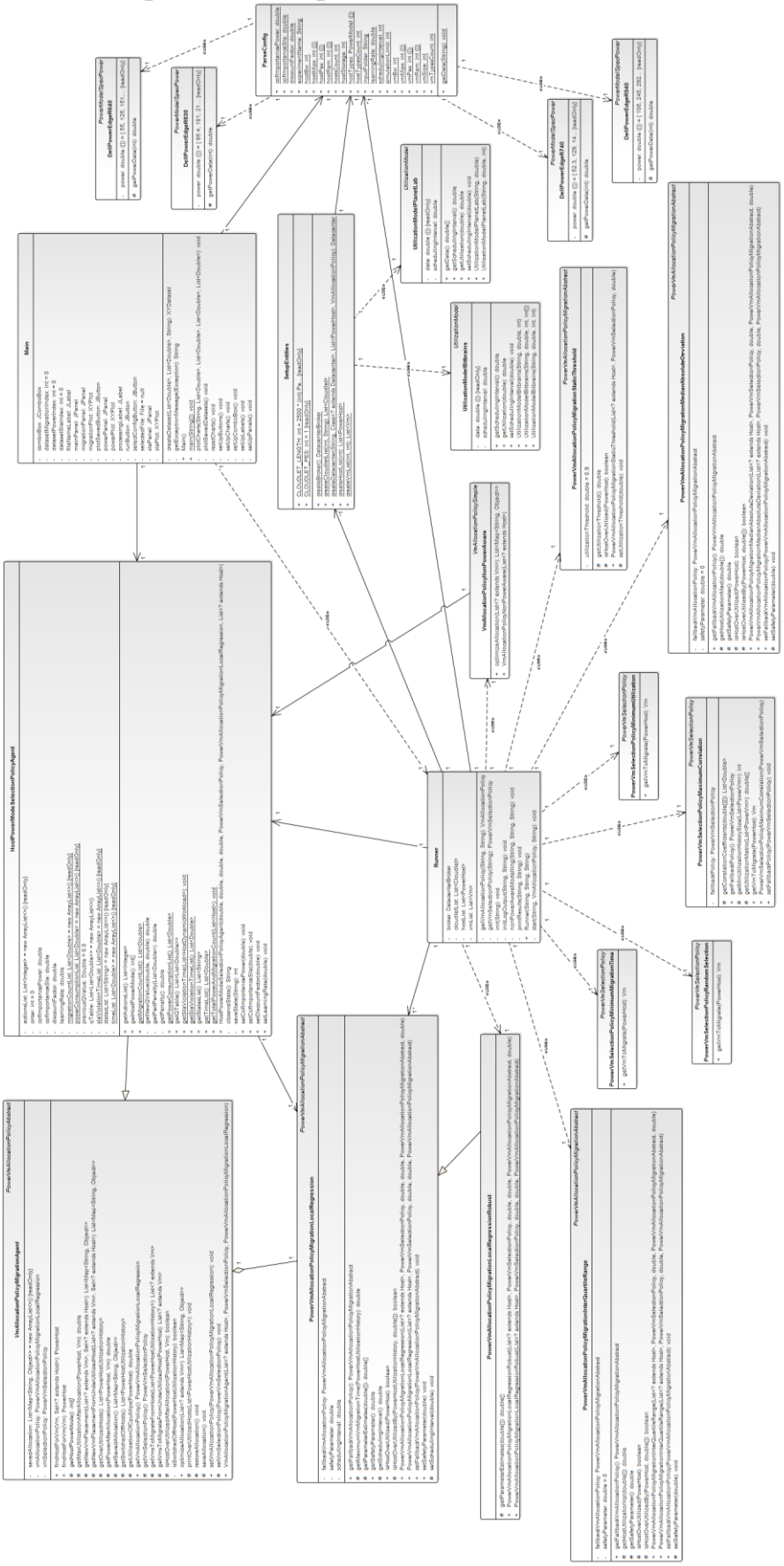
Представлення класу-фасаду



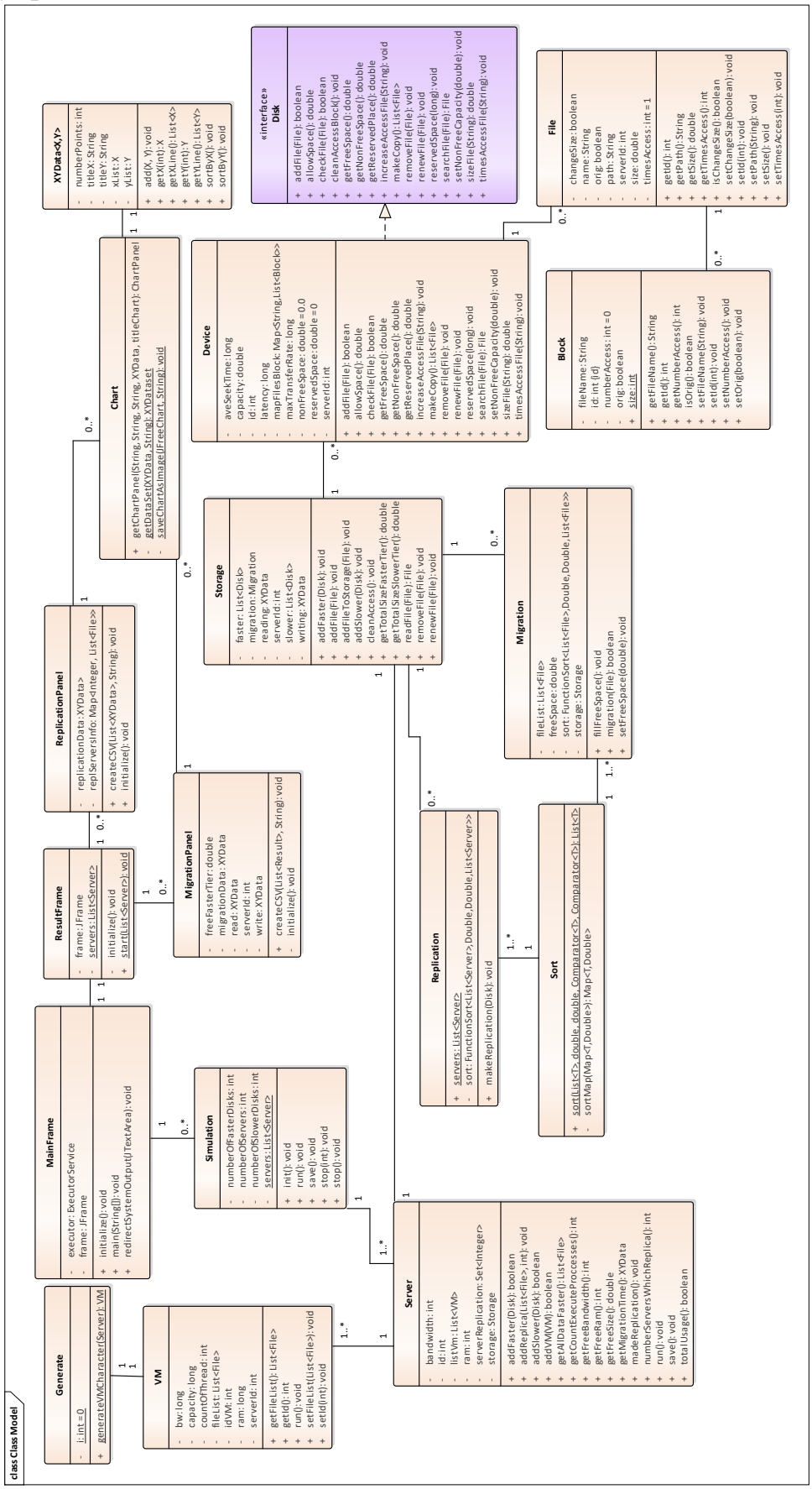
Представлення класів сутностей бази даних



Діаграма класів для моделей та компонентів методу динамічної консолідації і розміщення ВМ на основі модифікованого методу навчання з підкріпленням в схемі реалізації стратегій S_5 і S_6.



Діаграма класів для моделей та компонентів методу управління реплікацією та міжрівневою міграцією даних у сховищі хмарного ЦОД у схемі реалізації стратегій управління.



ДОДАТОК Б

Код адаптивного методу комбінованого прогнозування у схемі реалізації стратегій управління.

```
CN =
c("Timestamp","CPUcores","CPUcapprov","CPUusage","CPUusageP","Mem_prov_KB","Mem_
usage_KB","Disk_read_KBps","Disk_write_KBps","Network_received_KBps","Network_tr
_KBps")
vmpath = "C:/Apps/DEV/TRACES/GWA-T-12 Bitbrains/fastStorage/2013-8/"
logpath = "C:/Apps/DEV/TRACES/GWA-T-12 Bitbrains/fastStorage/"
require(forecast)
N=1152        # 2016*3 7 діб з інтервалом 5 хвилин, 3 тижні. 288 - 1 доба, 1152
- 4 доби
minSW = 4       # sliding window min size
maxSW = 80       # sliding window max size
horizon = 1         # forecast horizon
resfile = ''

for (vmnum in c(850, 795, 840, 599, 904, 957)) {
  fname = paste(logpath,"NEW-METHOD-5FMet","-",N,"-","VM904_W4-80","-CPU.csv",
sep="")
  vmname = paste(vmnum,".csv", sep="")
  vmfullname = paste(vmpath, vmname, sep="")
  vm <- read.csv(vmfullname, nrows = N, header = T, sep = ";",  col.names = CN)
  CPU_ts = ts(vm[,5])
  fstep = maxSW
  minW1p = ''
  minM1p = ''
  minW2p = ''
  minM2p = ''
  minW3p = ''
  minM3p = ''
  minW4p = ''
  minM4p = ''
  minW5p = ''
  minM5p = ''
  while (fstep < N) {
    setoffcst = ''
    testts <- window(CPU_ts,start=fstep+1,end=fstep+horizon)
    for (i in minSW:maxSW) {
      ts <- window(CPU_ts,start=fstep-i+1,end=fstep)

      # ARIMA i auto.arima
      ARIMAfit = auto.arima(ts,lambda=0, biasadj=TRUE)
      arimafcst = forecast(ARIMAfit, h=horizon, level=95)
      ACarima = accuracy(arimafcst,testts)
      #RV - real value, FCV forecasted value MAPE - похибка прогнозу
      newRow <- data.frame(Window = i, Method='ARIMA', RV =
      as.numeric(testts[1]), FCV = as.numeric(arimafcst$mean[1]),
      MAPE=as.numeric(ACarima[10]))
      if (i == minSW) (setoffcst <- newRow) else (setoffcst <-
      rbind(setoffcst,newRow))

      # SES HOLT
      fitses <- ses(ts, h=horizon)
      fitholt <- holt(ts, h=horizon)
      fitdholt <- holt(ts, damped=TRUE, phi = 0.9, h=horizon)
      ACses=accuracy(fitses$mean[1],testts)
      ACholt=accuracy(fitholt$mean[1],testts)
      ACdholt=accuracy(fitdholt$mean[1],testts)
      newRow <- data.frame(Window = i, Method='SES', RV = as.numeric(testts[1]),
      FCV = as.numeric(fitses$mean[1]), MAPE=as.numeric(ACses[5]))
```

```
      setoffcst <- rbind(setoffcst,newRow)
      newRow <- data.frame(Window = i, Method='HOLT', RV =
      as.numeric(testts[1]), FCV = as.numeric(fitholt$mean[1]),
      MAPE=as.numeric(ACholt[5]))
      setoffcst <- rbind(setoffcst,newRow)
      newRow <- data.frame(Window = i, Method='DHOLT', RV =
      as.numeric(testts[1]), FCV = as.numeric(fitdholt$mean[1]),
      MAPE=as.numeric(ACdholt[5]))
      setoffcst <- rbind(setoffcst,newRow)

      # LRtrend
      fitLRt <- tslm(ts ~ trend)
      fcLRt <- forecast(fitLRt,horizon)
      ACLRt=accuracy(fcLRt$mean[1],testts)
      newRow <- data.frame(Window = i, Method='LRT', RV = as.numeric(testts[1]),
FCV = as.numeric(fcLRt$mean[1]), MAPE=as.numeric(ACLRt[5]))
      setoffcst <- rbind(setoffcst,newRow)

      # HOLT-Winters (HW)
      fithw <- hw(ts, seasonal="multiplicative", h=horizon)
      ACholt=accuracy(fithw$mean[1],testts)
      newRow <- data.frame(Window = i, Method='HW', RV = as.numeric(testts[1]),
      FCV = as.numeric(fcTBATS$mean[1]), MAPE=as.numeric(ACtbats[5]))
      setoffcst <- rbind(setoffcst,newRow)
      # TBATS
      fcTBATS <- forecast(tbats(ts, biasadj=TRUE), h=horizon)
      ACtbats=accuracy(fcTBATS[["mean"]],testts)
      newRow <- data.frame(Window = i, Method='TBATS', RV =
      as.numeric(testts[1]), FCV = as.numeric(fcTBATS$mean[1]),
      MAPE=as.numeric(ACtbats[5]))
      setoffcst <- rbind(setoffcst,newRow)
    }
    sortedfcst <- setoffcst[order(setoffcst$MAPE),]
    minW1 = sortedfcst[1,"Window"]
    minM1 = sortedfcst[1,"Method"]
    minW2 = sortedfcst[2,"Window"]
    minM2 = sortedfcst[2,"Method"]
    minW3 = sortedfcst[3,"Window"]
    minM3 = sortedfcst[3,"Method"]
    minW4 = sortedfcst[4,"Window"]
    minM4 = sortedfcst[4,"Method"]
    minW5 = sortedfcst[5,"Window"]
    minM5 = sortedfcst[5,"Method"]
    if (fstep == maxSW) {
      NR <- data.frame(VmNumber=vmnum, Step=fstep, RealValue =
      as.numeric(testts[1]),
                       W1=minW1, M1=minM1, FCV1 = sortedfcst[1,"FCV"],
                       MAPE1=sortedfcst[1,"MAPE"],
                       W2=minW2, M2=minM2, FCV2 = sortedfcst[2,"FCV"],
                       MAPE2=sortedfcst[2,"MAPE"],
                       W3=minW3, M3=minM3, FCV3 = sortedfcst[3,"FCV"],
                       MAPE3=sortedfcst[3,"MAPE"],
                       W4=minW4, M4=minM4, FCV4 = sortedfcst[4,"FCV"],
                       MAPE4=sortedfcst[4,"MAPE"],
                       W5=minW5, M5=minM5, FCV5 = sortedfcst[5,"FCV"],
                       MAPE5=sortedfcst[5,"MAPE"],
                       W1p=NA, M1p=NA, FCV1p = NA, MAPE1p=NA,
                       W2p=NA, M2p=NA, FCV2p = NA, MAPE2p=NA,
                       W3p=NA, M3p=NA, FCV3p = NA, MAPE3p=NA,
                       W4p=NA, M4p=NA, FCV4p = NA, MAPE4p=NA,
                       W5p=NA, M5p=NA, FCV5p = NA, MAPE5p=NA)
    } else {
      #minW1p=setoffcst[2,1]
      #minM1p=setoffcst[2,2]
```

```
      #CDpi1 <- subset(setoffcst, Window == minW1p, Method == minM1p, select = )
      CDpi1 <- setoffcst[which(setoffcst$Window == minW1p & setoffcst$Method ==
      minM1p),]
      CDpi2 <- setoffcst[which(setoffcst$Window == minW2p & setoffcst$Method ==
      minM2p),]
      CDpi3 <- setoffcst[which(setoffcst$Window == minW3p & setoffcst$Method ==
      minM3p),]
      CDpi4 <- setoffcst[which(setoffcst$Window == minW4p & setoffcst$Method ==
      minM4p),]
      CDpi5 <- setoffcst[which(setoffcst$Window == minW5p & setoffcst$Method ==
      minM5p),]
      NR <- data.frame(VmNumber=vmnum, Step=fstep, RealValue =
      as.numeric(testts[1]),
      W1=minW1, M1=minM1, FCV1 = sortedfcst[1,"FCV"],
      MAPE1=sortedfcst[1,"MAPE"],
     W2=minW2, M2=minM2, FCV2 = sortedfcst[2,"FCV"], MAPE2=sortedfcst[2,"MAPE"],
     W3=minW3, M3=minM3, FCV3 = sortedfcst[3,"FCV"], MAPE3=sortedfcst[3,"MAPE"],
     W4=minW4, M4=minM4, FCV4 = sortedfcst[4,"FCV"], MAPE4=sortedfcst[4,"MAPE"],
     W5=minW5, M5=minM5, FCV5 = sortedfcst[5,"FCV"], MAPE5=sortedfcst[5,"MAPE"],
      W1p=CDpi1[1,"Window"], M1p=CDpi1[1,"Method"], FCV1p = CDpi1[1,"FCV"],
      MAPE1p=CDpi1[1,"MAPE"],
     W2p=CDpi2[1,"Window"], M2p=CDpi2[1,"Method"], FCV2p = CDpi2[1,"FCV"],
     MAPE2p=CDpi2[1,"MAPE"],
     W3p=CDpi3[1,"Window"], M3p=CDpi3[1,"Method"], FCV3p = CDpi3[1,"FCV"],
     MAPE3p=CDpi3[1,"MAPE"],
     W4p=CDpi4[1,"Window"], M4p=CDpi4[1,"Method"], FCV4p = CDpi4[1,"FCV"],
     MAPE4p=CDpi4[1,"MAPE"],
     W5p=CDpi5[1,"Window"], M5p=CDpi5[1,"Method"], FCV5p = CDpi5[1,"FCV"],
     MAPE5p=CDpi5[1,"MAPE"])
    }
    minW1p = minW1
    minM1p = minM1
    minW2p = minW2
    minM2p = minM2
    minW3p = minW3
    minM3p = minM3
    minW4p = minW4
    minM4p = minM4
    minW5p = minW5
    minM5p = minM5
    if ((vmnum == 904) & (fstep == maxSW)) (resfile <- NR) else (resfile <-
rbind(resfile,NR))
    fstep = fstep+1
  }
}
write.csv(resfile, file = fname)
```

ДОДАТОК В

Впровадження результатів дисертації

ТОВ «АМ ІНТЕГРАТОР ГРУП»
04070, м. Київ, вул. Братська, буд. 6.
тел.: +38 (044) 394-86-07, 394-86-09
mail@amintegrator.com, http://amintegrator.com

Вих.№ 328 від 8 квітня 2019 р.

ДОВІДКА

про впровадження результатів дисертаційної роботи

Жарікова Едуарда В'ячеславовича, поданої на здобуття наукового ступеня доктора технічних наук, на тему «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних»

при розробленні системи управління функціонуванням інформаційно-телекомунікаційної інфраструктури ТОВ «АМ ІНТЕГРАТОР ГРУП»

Результати дисертаційної роботи Жарікова Е. В. у вигляді інформаційної технології управління ІТ-інфраструктурою центру оброблення даних використані при створенні системи управління функціонуванням інформаційно-телекомунікаційної інфраструктури ТОВ «АМ ІНТЕГРАТОР ГРУП».

1. Використання запропонованих в дисертаційній роботі принципів побудови програмно-визначеної системи управління ресурсами ІТ-інфраструктури, моделей і методів інтегрованого управління ресурсами з використанням прогнозування дозволили на 28% скоротити операційні витрати на управління ІТ-інфраструктурою.

2. Моделі і методи управління ресурсами на основі променевого пошуку та навчання з підкріпленням використані при створенні підсистеми управління фізичними серверами, що є складовою системи управління функціонуванням інформаційно-телекомунікаційної інфраструктури ТОВ «АМ ІНТЕГРАТОР ГРУП».

3. Підсистема управління ресурсами з використанням запропонованого в роботі методу на основі навчання з підкріпленням дозволяє підтримувати заданий рівень надання послуг при змінному комбінованому навантаженні користувачів. Завдяки використанню запропонованого в дисертаційній роботі

адаптивного методу прогнозування навантаження на обчислювальні ресурси хмарного ЦОД з використанням усередненого і зваженого комбінування альтернативних методів прогнозування зменшилася кількість порушень SLA на 28%.

4. Застосування методу управління потужністю ЦОД разом з метриками відстеження стану ресурсів дозволили зменшити споживання електроенергії на 17% в середовищі з гомогенними конфігураціями фізичних серверів.

Голова наглядової ради Брейман М.Г.

"АМ ІНТЕГРАТОР ГРУП" Ідентифікаційний код 39469334

ТОВ "СІТІУС ПРО"
04050, Київ, Мельникова 81А
Тел: +380 (44) 390-88-18
E-Mail: Office@Citius.pro

Вих. № 189 Від 06 Квітня 2019 р.

Довідка

про впровадження результатів дисертаційної роботи
Жарікова Едуарда В'ячеславовича, поданої на здобуття наукового ступеня доктора технічних наук, на тему «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних»
при розробленні системи управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО»

Інформаційна технологія управління ІТ-інфраструктурою, що розроблена в дисертаційній роботі Жарікова Е. В., використана при створенні системи управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО». Впровадження положень дисертаційної роботи дало такі результати:

1. Розроблена здобувачем інформаційна технологія управління ІТ-інфраструктурою використана при модернізації системи управління як сукупність методів та принципів ефективного управління ресурсами, якістю послуг та автоматизації оперативного управління парком фізичних серверів в ІТ-інфраструктурі ТОВ «СІТІУС ПРО».

2. Запропоновані методи і алгоритми управління віртуальними машинами і фізичними серверами використані при розробці системного програмного забезпечення управління ІТ-інфраструктурою. Експлуатаційні випробування продемонстрували гнучкість при налаштуванні серверного обладнання системи та зменшення кількості порушень вимог угоди про рівень обслуговування при реалізації інформаційних процесів.

3. Запропоновані в роботі методи прогнозування, метрики оцінювання стану фізичних серверів і віртуальних машин, алгоритм управління міграцією і розміщенням нових ВМ в режимі онлайн заклали методологічні основи побудови підсистем управління гіпервізорами фізичних серверів в системі управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО». Це дозволило скоротити в середньому на 18% кількість фізичних серверів, що обслуговують навантаження клієнтів.

4. Завдяки використанню запропонованих в дисертації методів, моделей та алгоритмів оперативного управління ресурсами, навантаженням і

ТОВ "СІТІУС ПРО"
04050, Київ, Мельникова 81А
Тел: +380 (44) 390-88-18
E-Mail: Office@Citius.pro

потужністю з використанням прогнозування кількість порушень SLA при наданні IT-послуг зменшилася на 27%.

5. Використання запропонованих в дисертаційній роботі моделей і методів управління ресурсами в умовах експлуатації гіперконвергентних і програмно-визначених систем, що є складовою IT-інфраструктури ТОВ «СІТІУС ПРО», дозволило скоротити витрати на експлуатацію серверного парку на 19% при забезпеченні виконання заданих вимог угоди про рівень обслуговування.

Директор ТОВ «СІТІУС ПРО» Гомлявий О.В.

ТЕЛАС
УКРАЇНСЬКА АСОЦІАЦІЯ ОПЕРАТОРІВ ЗВ'ЯЗКУ

вул. Солом'янська, 3, оф.808, м. Київ, 03110, Україна, тел/факс 248 9171, 248 9175
www.telas.kiev.ua **e-mail: astelas@ukrpack.net**

вих. № 40
від 06 травня 2019 року

Довідка

про впровадження результатів дисертаційної роботи
Жарікова Едуарда В'ячеславовича, поданої на здобуття наукового ступеня
доктора технічних наук, на тему «Інформаційна технологія управління IT-
інфраструктурою хмарного центру оброблення даних»
у процесах діяльності компаній-членів Асоціації «ТЕЛАС»

Результати дисертаційної роботи Жарікова Е. В. у вигляді інформаційної технології управління IT-інфраструктурою центру оброблення даних використані при модернізації систем управління функціонуванням інформаційно-телекомунікаційної інфраструктури в окремих операторів телекомунікацій, які входять до складу Української асоціації операторів зв'язку «ТЕЛАС».

1. Запропонована в роботі модель багаторівневої ієрархічної системи управління IT-інфраструктурою хмарного ЦОД, яка використовує набори різних параметрів, визначених в хмарному ЦОД, таких як: набір послуг, набір управлінських впливів на кожному шарі моделі, набір показників якості обслуговування і набір споживаних ресурсів заклала методологічну основу побудови підсистем управління та додаткових політик управління процесами створення і функціонування екземплярів класів на кожному рівні ієрархії (застосунків, платформ і інфраструктури). Реалізація запропонованого в дисертаційній роботі метода управління на основі моделі багаторівневої ієрархічної системи управління IT-інфраструктурою хмарного ЦОД дала можливість зменшити складність скриптів управління елементами IT-інфраструктури, більш чітко відокремити зони відповідальності адміністраторів та зменшити витрати за порушення SLA в середньому на 6%.

2. Запропоновані в роботі моделі усередненого комбінованого прогнозування і зваженого комбінованого прогнозування використані в системах управління фізичними серверами в умовах хмарних обчислень де визначення керуючих впливів на основі прогнозних значень дозволило зменшити енергоспоживання на 9%.

3. Запропоновані в роботі методи і алгоритми стохастичного пошуку використані для визначення керуючих впливів за умов довільної кількості ресурсів гетерогенної фізичної інфраструктури, що дозволило підвищити якість надання хмарних послуг в режимах роботи ЦОД з високою дисперсією навантаження, і при цьому зменшити кількість працюючих фізичних серверів в середньому на 5%.

4. Впровадження методу управління реплікацією та міжрівневою міграцією даних у сховищах ЦОД в системах з гіперконвергентною архітектурою дозволило більш рівномірно завантажити вузли збереження даних і підвищити продуктивність роботи вузлів обробки даних за рахунок зменшення впливу процесу реплікації даних.

Голова Ради Української асоціації операторів зв'язку «ТЕЛАС» Леонід Ошеров

Українська асоціація операторів зв'язку * Україна * м.Київ * «ТЕЛАС» Ідентифікаційний код № 24725145

"ЗАТВЕРДЖУЮ"
Перший заступник декана
факультету інформатики та
обчислювальної техніки
КПІ ім. Ігоря Сікорського
к.т.н., доц. Андрій ПИСАРЕНКО

"20" грудня 2019 р.

АКТ

впровадження у навчальний процес результатів дисертаційної роботи доцента кафедри автоматизованих систем обробки інформації і управління Жарікова Едуарда В'ячеславовича, поданої на здобуття наукового ступеня доктора технічних наук, на тему "Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних"

Комісія у складі: голова – завідуючий кафедрою автоматизованих систем обробки інформації і управління, проф. Павлов О. А.; члени комісії – доц. Муха І. П., доц. Ліщук К. І., доц. Гавриленко О. В. підтвердила, що на кафедрі автоматизованих систем обробки інформації і управління КПІ ім. Ігоря Сікорського впроваджені результати дисертаційної роботи доц. Жарікова Е. В. в навчальних дисциплінах "Технології віртуалізації та хмарних обчислень", "Сучасні технології розроблення програмного забезпечення", "Інтелектуальні системи управління технічними пристроями", "Програмування інтернету речей", "Методи та системи штучного інтелекту", "Обробка надвеликих масивів інформації".

Впровадження результатів дисертаційної роботи доц. Жарікова Е. В. дає можливість ознайомити студентів з новими підходами і технологіями управління ІТ-інфраструктурою хмарного центру оброблення даних та дозволяє розробляти власні методи і алгоритми управління ресурсами для сучасних віртуалізованих та програмно-визначених програмно-апаратних комплексів.

Голова комісії _____ проф. Олександр ПАВЛОВ

Члени комісії: _____ доц. Ірина МУХА

_____ доц. Катерина ЛІЩУК

_____ доц. Олена ГАВРИЛЕНКО

ДОДАТОК Г

Апробація результатів дисертації

ВИТЯГ

з протоколу № 9 від 03.12.2019 р.
засідання наукового семінару Інституту прикладного системного аналізу
Національного технічного університету України
«Київський політехнічний інститут імені Ігоря Сікорського»

1. ПРИСУТНІ: 12 членів семінару: д.т.н. проф. Данилов В.Я., д.т.н. проф. Бідюк П.І., д.т.н. проф. Андрєєв М.В., д.т.н. проф. Зайченко О.Ю., д.т.н. проф. Снитюк В.Є., д.ф.-м.н. проф. Бондаренко В.Г., к.т.н. доц. Коваленко А.Є., к.т.н. доц. Терентьєв О.М., к.т.н. доц. Савченко І.О., к.т.н. доц. Жиров О.Л., к.т.н. доц. Брітов О.А., к.т.н. доц. Булах Б.В.

2. СЛУХАЛИ:

2.1. Доповідь доцента кафедри автоматизованих систем обробки інформації і управління КПІ ім. Ігоря Сікорського за матеріалами дисертаційної роботи на тему: «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних», представленої на здобуття наукового ступеня доктора технічних наук за спеціальністю 05.13.06 – інформаційні технології.

Науковий консультант – доктор технічних наук, професор, декан факультету інформатики та обчислювальної техніки КПІ ім. Ігоря Сікорського Теленик С.Ф.

2.2. Питання та відповіді по доповіді.

По доповіді було задано 17 запитань, на які доповідач дав правильні і ґрунтовні відповіді. Питання були задані: д.т.н. проф. Бідюком П.І., д.т.н. проф. Даниловим В.Я., д.т.н. проф. Романенком В.Д., д.т.н. проф. Зайченко О.Ю., д.т.н. проф. Снитюком В.Є. та іншими членами наукового семінару.

2.3. При обговоренні роботи виступили 5 присутніх.

З позитивною оцінкою дисертаційної роботи Жарікова Е.В. виступили:

- д.т.н. проф. Бідюк П.І., який зауважив, що сформульовані наукові положення дисертаційного дослідження ще вимагають узгодження та корегування, а також запропонував довести теорему про функціональну повноту системи керування;
- д.т.н. проф. Данилов В.Я., який запропонував конкретизувати об'єкт керування як нелінійний нестаціонарний дискретний об'єкт зі змінною структурою, а також рекомендував здобувачеві уточнити формулювання критеріїв якості моделі двостадійного методу управління ресурсами хмарного ЦОД на основі алгоритму променевого пошуку;
- д.т.н. проф. Зайченко О.Ю., яка у якості зауважень відмітила застосування автором окремих некоректних або невдалих термінів (наприклад «визначення коефіцієнтів моделі», «великий розмір задачі», «центр обробки даних» за авторефератом), зазначила необхідність більш досконалого обґрунтування застосування операторної форми постановки, аналізу і розв'язання задач управління ІТ-інфраструктурою хмарного ЦОД;
- д.ф.-м.н. проф. Андрєєв М.В., який запропонував додатково зробити огляд наукових робіт на предмет врахування часових інтервалів міграції віртуальних машин в мережах ЦОД з різними швидкісними показниками;
- д.т.н. проф. Снитюк В.Є., який запропонував шляхи ідентифікації та усунення невизначеності в процесі керування в умовах змінних навантажень на ресурси ІТ-інфраструктури.

Присутні на засіданні обговорили проект висновку, підготовлений д.т.н. проф. Бідюком П. І.

3. Заслухавши та обговоривши доповідь здобувача прийняли до уваги такі висновки щодо попереднього розгляду дисертаційної роботи Жарікова Е.В.

В результаті експертизи дисертаційної роботи, представленої на семінарі Інституту прикладного системного аналізу Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» встановлено:

- в ході обговорення дисертаційної роботи до неї не було висунуто зауважень принципового характеру, що торкаються самої суті роботи. Дисертація містить нові, раніше незахищені наукові положення та отримані автором нові науково обґрунтовані результати у обраній галузі науки і техніки, які у сукупності розв'язують важливу науково-прикладну проблему;

- дисертація була кваліфікована як закінчена наукова робота та рекомендована до захисту на засіданні спеціалізованої вченої ради за спеціальністю 05.13.06 – інформаційні технології.

4. СЕМІНАР УХВАЛИВ:

1. Дисертаційна робота Жарікова Е.В. на тему «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних» є закінченою науковою працею, у якій розв'язано важливу наукову проблему забезпечення ефективного функціонування ІТ-інфраструктури хмарних центрів оброблення даних за рахунок розроблення та застосування інформаційної технології управління ІТ-інфраструктурою, яка дозволяє забезпечити виконання заданих вимог угоди про рівень обслуговування та зниження операційних і капітальних витрат, що має важливе значення при наданні інформаційних послуг сучасними центрами оброблення даних.

2. У 70 наукових публікаціях автора повністю висвітлені матеріали дисертаційного дослідження здобувача. З них 17 публікацій у фахових наукових виданнях України та 10 публікацій у закордонних наукових виданнях.

3. Дисертаційна робота Жарікова Е.В. на тему «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних» повністю відповідає паспорту спеціальності 05.13.06 – інформаційні технології і пп. 9, 11 «Порядку присудження наукових ступенів» затвердженого постановою Кабінету Міністрів України від 24 липня 2013 р. № 567.

4. З врахуванням наукової зрілості та професійних якостей дисертанта Жарікова Е.В. його дисертаційна робота «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних» рекомендується для подання до захисту у спеціалізовану вчену раду.

Результати голосування за висновок по дисертаційній роботі:
за – дванадцять членів семінару;
проти – немає;
утримались – немає.

Голова засідання:
д.т.н., проф. В. Я. Данилов

Вчений секретар:
д.т.н., проф. П. І. Бідюк

ВИТЯГ

з протоколу № 7 від 18.12.2019 р.
засідання наукового семінару кафедри інтелектуальних технологій
факультету інформаційних технологій
Київського національного університету імені Тараса Шевченка

1. ПРИСУТНІ: 11 членів семінару: д.т.н. проф. Снитюк В. Є., д.т.н. проф. Циганок В. В., д.т.н. проф. Кудін В. І., к.т.н., проф. Гайна Г.А., к.т.н. доц. Іларіонов О. Є., к.ф.-м.н. доц. П.М.Сорока, к.т.н. доц. Красовська Г.В., к.т.н. доц. Федусенко О.В., к.т.н. доц. Доманецька І.М., к.т.н. доц. Кіктєв М.О., к.т.н. Гнатієнко Г.М.

2. СЛУХАЛИ:

2.1. Доповідь доцента кафедри автоматизованих систем обробки інформації і управління КПІ ім. Ігоря Сікорського, Жарікова Едуарда В'ячеславовича за матеріалами дисертаційної роботи на тему: «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних», представленої на здобуття наукового ступеня доктора технічних наук за спеціальністю 05.13.06 – інформаційні технології.

Науковий консультант – доктор технічних наук, професор, декан факультету інформатики та обчислювальної техніки КПІ ім. Ігоря Сікорського Теленик С. Ф.

2.2. Питання та відповіді по доповіді.

По доповіді було задано 15 запитань, на які доповідач дав правильні і ґрунтовні відповіді. Питання були задані: д.т.н. проф. Снитюком В. Є., д.т.н. проф. Циганком В. В., д.т.н. проф. Кудіним В. І., к.т.н. проф. Гайна Г.А., та іншими членами наукового семінару.

2.3. При обговоренні роботи виступили 4 присутніх.

З позитивною оцінкою дисертаційної роботи Жарікова Е.В. виступили:

- д.т.н. проф. Снитюк В. Є., який зауважив, що наукові положення дисертаційного дослідження, що стосуються інформаційної технології і операторної форми постановки, аналізу і розв'язання задач управління ІТ-інфраструктурою хмарного ЦОД ще вимагають корегування, а також зазначив необхідність кількісних показників у висновках до дисертації;
- д.т.н. проф. Циганок В. В., який рекомендував здобувачеві уточнити формулювання деяких пунктів новизни та уточнити критерій вибору стратегії управління ресурсами ІТ-інфраструктури хмарного ЦОД, а також відмітив застосування автором окремих невдалих термінів, наприклад «тренувальні дані», «історичні дані», «множина параметрів методу» за авторефератом);
- д.т.н. проф. Кудін В. І., який рекомендував здобувачеві уточнити зв'язок критерію вибору стратегії управління ресурсами ІТ-інфраструктури з операторною формою постановки, аналізу і розв'язання задач управління ІТ-інфраструктурою;
- к.т.н. доц. Іларіонов О. Є., який запропонував додатково зробити огляд наукових робіт, які стосуються комбінування прогнозів, отриманих набором альтернативних методів і моделей прогнозування;

Присутні на засіданні обговорили проект висновку, підготовлений к.т.н. доц. Іларіоновим О. Є.

3. Заслухавши та обговоривши доповідь здобувача прийняли до уваги такі висновки щодо попереднього розгляду дисертаційної роботи Жарікова Е. В.

В результаті експертизи дисертаційної роботи, представленої на семінарі кафедри інтелектуальних технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка встановлено:

- в ході обговорення дисертаційної роботи до неї не було висунуто зауважень принципового характеру, що стосуються самої суті роботи. Дисертація містить нові, раніше незахищені наукові положення та отримані автором нові науково обґрунтовані результати у обраній галузі науки і техніки, які у сукупності розв'язують важливу науково-прикладну проблему.

- дисертація була кваліфікована як закінчена наукова робота та рекомендована до захисту на засіданні спеціалізованої вченої ради за спеціальністю 05.13.06 – інформаційні технології.

4. СЕМІНАР УХВАЛИВ:

1. Дисертаційна робота Жарікова Е. В. на тему «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних» є закінченою науковою працею, у якій вирішено важливу науково-практичну проблему забезпечення ефективного функціонування ІТ-інфраструктури хмарних ЦОД шляхом розроблення і застосування інформаційної технології управління ресурсами, що дозволяє забезпечити зменшення операційних витрат та надання хмарних послуг із заданими показниками якості кінцевому користувачеві.

2. У 70 наукових публікаціях автора повністю висвітлені матеріали дисертаційного дослідження здобувача. З них 17 публікацій у фахових наукових виданнях України та 10 публікацій у закордонних наукових виданнях.

3. Дисертаційна робота Жарікова Е.В. на тему «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних» відповідає паспорту спеціальності 05.13.06 – інформаційні технології і пп. 9, 11 «Порядку присудження наукових ступенів» затвердженого постановою Кабінету Міністрів України від 24 липня 2013 р. № 567.

4. З врахуванням наукової зрілості та професійних якостей дисертанта Жарікова Е. В. його дисертаційна робота «Інформаційна технологія управління ІТ-інфраструктурою хмарного центру оброблення даних» може бути подана до захисту у спеціалізовану вчену раду.

Результати голосування за висновок по дисертаційній роботі:
за – одинадцять членів семінару;
проти – немає;
утримались – немає.

Голова засідання:
д.т.н., проф. Снитюк В.Є.

Вчений секретар:
к.т.н., доц. Іларіонов О.Є.